Quan Z. Sheng
Guoren Wang
Christian S. Jensen
Guandong Xu (Eds.)

LNCS 7235

Web Technologies and Applications

14th Asia-Pacific Web Conference, APWeb 2012
Kunming, China, April 2012
Proceedings

Springer

Lecture Notes in Computer Science 7235

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Quan Z. Sheng   Guoren Wang
Christian S. Jensen   Guandong Xu (Eds.)

Web Technologies
and Applications

14th Asia-Pacific Web Conference, APWeb 2012
Kunming, China, April 11-13, 2012
Proceedings

Springer

Volume Editors

Quan Z. Sheng
The University of Adelaide, School of Computer Science
Adelaide, SA 5005, Australia
E-mail: qsheng@cs.adelaide.edu.au

Guoren Wang
Northeastern University, College of Information Science and Engineering
Shenyang 110819, China
E-mail: wanggr@mail.neu.edu.cn

Christian S. Jensen
Aarhus University, Department of Computer Science
8200 Aarhus N, Denmark
E-mail: csj@cs.au.dk

Guandong Xu
Victoria University, Centre for Applied Informatics
Melbourne City, VIC 8001, Australia
E-mail: guandong.xu@vu.edu.au

Message from the General Chairs

Welcome to APWeb 2012, the 14th Edition of Asia Pacific Web Technology Conference. APWeb is a leading international conference on research, development, and applications of Web technologies, database systems, information management, and software engineering, with a focus on the Asia-Pacific region. Previous APWeb conferences were held in Beijing (2011), Busan (2010), Suzhou (2009), Shenyang (2008), Huangshan (2007), Harbin (2006), Shanghai (2005), Hangzhou (2004), Xi'an (2003), Changsha (2001), Xi'an (2000), Hong Kong (1999), and Beijing (1998).

APWeb conferences cover contemporary topics in the fields of Web management and World Wide Web-related research and applications, such as advanced application of databases, cloud computing, content management, data mining and knowledge discovery, distributed and parallel processing, grid computing, Internet of Things, Semantic Web and Web ontology, security, privacy and trust, sensor networks, service-oriented computing, Web community analysis, Web mining and social networks.

The APWeb 2012 program featured a main conference and five satellite workshops. The main conference had two eminent keynote speakers, Patrick McDaniel from Pennsylvania State University, USA, and Vijay Varadharajan from Macquarie University, Australia, 39 full research papers, 34 short research papers, and 5 demonstration papers. The four workshops reported recent developments and advances in the contemporary topics in the related fields of:

- Information Extraction and Knowledge Base Building (IEKB)
- Intelligent Data Processing (IDP 2012)
- Sensor Networks and Data Engineering (SenDe 2012)
- Mobile Business Collaboration (MBC 2012)

Both main conference program and workshop sessions were of high quality owing to the strong support and commitment from their international Program Committee. We wish to take this opportunity to thank Program Co-chairs Michael Sheng, Guoren Wang, and Christian S. Jensen, for their dedication and effort in ensuring a high-quality program. We would also like to thank Workshop Co-chairs Hua Wang and Lei Zou, and each workshop organizers for their contributions to developing an interesting and attractive workshop program. Many colleagues helped toward the success of APWeb 2012. They are: Local Arrangements Co-chairs: Jianfeng He and Guangyan Huang; Financial Chair: Jing He; Publication Chair: Guandong Xu; Industry Co-chairs: Gary Morgan and Qingzhong Li; Demonstration Chair: Chaoyi Pang; Publicity Co-chairs: Haolan Zhang and Jiangang Ma, and Webmaster: Zhi Qiao and Zhangwei Jiang.

We would like to sincerely thank our financial supporters and sponsors. The following organizations generously supported and sponsored APWeb 2012:

Hebei University of Engineering, Nanjing University of Finance and Economics, National Science Foundation of China, Kunming University of Science and Technology, Graduate University of Chinese Academy of Science, and Victoria University.

Finally, we wish to thank the APWeb Steering Committee, led by Xuemin Lin, for offering the opportunity to organize APWeb 2012 in Kunming. We also wish to thank the host organization Kunming University of Science and Technology and local arrangements committee and volunteers for the assistance in organizing this conference. The following members helped with the registration, accommodation, and various logistics: Jing Yang, Xun Zhou, Fenhua Li, and Shang Hu.

February 2012   Yaoxue Zhang
                                                        Yanchun Zhang
                                                    Masaru Kitsuregawa

Message from the Program Chairs

The relentless growth in Internet functionality and broadband access has enabled a new wave of innovations that is transforming the way people and organizations interact, communicate, and collaborate. APWeb 2012 built on the tradition of 13 successful previous editions of the conference and aimed at presenting original contributions related to all fields of World Wide Web-related research, development, and applications.

The call for papers generated substantial interest. A total of 167 paper submissions were received, which represents an increase of over 60% compared to APWeb 2011. The paper submissions, from 17 countries and regions, were reviewed carefully by members of the Program Committee, which comprised 104 experts from 20 countries. Based on the reviews, 39 submissions were selected as full papers, yielding an acceptance rate of 23%. In addition, 34 submissions were selected as short papers. The selected papers covered a wide variety of important topics related to the World Wide Web, including Web mining, service-oriented computing, XML and semi-structured query processing, social networks, Web information extraction, and cloud computing. Authors of a few selected papers will be invited to submit extended versions to a special issue of the *World Wide Web* journal, published by Springer.

The conference program featured two keynote addresses given by Patrick Mc-Daniel, Editor-in-Chief of *ACM Transactions on Internet Technology*, and Vijay Varadharajan, Microsoft Chair in Innovation in Computing at Macquarie University, Australia. The program encompassed five demonstrations, which were selected by a separate committee chaired by Chaoyi Pang.

The program also contained a collection of papers that were presented in affiliated workshops and that were published in a separate LNCS volume, including the International Workshop on Intelligent Data Processing (IDP 2012) organized by Chaoyi Pang, Junhu Wang, and Haolan Zhang, the Workshop on Information Extraction and Knowledge Base Building (IEKB 2012) organized by Dongyan Zhao, the Workshop on Sensor Networks and Data Engineering (SenDe 2012) organized by Jing He, Guangyan Huang, and Xiedong Cao, and the Third International Workshop on Mobile Business Collaboration (MBC 2012) organized by Dickson Chiu, Yi Zhuang, and Zhiang Wu.

We thank all authors for their submissions and the Program Committee members and external reviewers for their excellent work. We hope that you find the papers in the proceedings interesting and stimulating.

February 2012

Quan Z. Sheng
Guoren Wang
Christian S. Jensen

Organization

Honorary General Chair

Yaoxue Zhang   Central South University, China

General Co-chairs

Yanchun Zhang   Victoria University, Australia
Masaru Kitsuregawa   The University of Tokyo, Japan

Program Co-chairs

Quan Z. Sheng   The University of Adelaide, Australia
Guoren Wang   Northeastern University, China
Christian S. Jensen   Aarhus University, Denmark

Industry Co-chairs

Gary Morgan   CSIRO, Australia
Qingzhong Li   Shandong University, China

Workshop Co-chairs

Hua Wang   University of Southern Queensland, Australia
Lei Zou   Peking University, China

Local Arrangements Chairs

Jianfeng He   Kunming University of Science and Technology, China
Guangyan Huang   Victoria University, Australia

Financial Chair

Jing He   Victoria University, Australia

Publication Chair

Guandong Xu   Victoria University, Australia

Demo Co-chair

Chaoyi Pang   Australia e-Health Research Centre, Australia

Publicity Co-chairs

Haolan Zhang   NIT, Zhejiang University, China
Jiangang Ma   The University of Adelaide, Australia

APWeb Steering Committee Liaison

Xuemin Lin   The University of New South Wales, Australia

WISE Steering Committee Liaison

Qing Li   City University of Hong Kong, China

Steering Committee

Xiaoming Li   Peking University, China
Xuemin Lin   University of New South Wales, Australia
Maria Orlowska   Ministry of Science and Higher Education,
                                            Poland
Kyu-Young Whang   Korea Advanced Institute of Science and
                                            Technology, Korea
Jeffrey Yu   Chinese University of Hong Kong, China
Yanchun Zhang   Victoria University, Australia
Xiaofang Zhou   University of Queensland, Australia

Program Committee

Toshiyuki Amagasa   University of Tsukuba, Japan
Denilson Barbosa   University of Alberta, Canada
Pablo Barcelo   Universidad de Chile, Chile
Djamal Benslimane   University of Lyon, France
Geert Jan Bex   Hasselt University, Belgium
Stephane Bressan   Singapore National University, Singapore
Jae-Woo Chang   Chonbuk National University, Korea
Wenguang Chen   Peking University, China
Mauro Caporuscio   Politecnico di Milano, Italy
Haiming Chen   Chinese Academy of Sciences, China
Hong Chen   Renmin University of China, China
Hanxiong Chen   University of Tsukuba, Japan
Jian Chen   Shaanxi Normal University, China
Jinchuan Chen   Renmin University of China, China

Jin Chen	Michigan State University, USA
Reynold Cheng	The University of Hong Kong, China
David Cheung	The University of Hong Kong, China
Richard Connor	University of Strathclyde, UK
Alfredo Cuzzocrea	ICAR-CNR and University of Calabria, Italy
Ke Deng	University of Queensland, Australia
Ting Deng	Beihang University, China
Paolo Falcarin	University of East London, UK
Ling Feng	Tsinghua University, China
Li Guo	Chinese Academy of Science, China
Xueqing Gong	East China Normal University, China
Rajeev Gupta	IBM Research, India
Qi He	IBM Research, USA
Sven Helmer	University of London, UK
Zi Huang	University of Queensland, Australia
Bin Hu	Lanzhou University, China
Yoshiharu Ishikawa	Nagoya University, Japan
Jin-Ho Kim	Kangwon National University, Korea
Kyong-Ho Lee	Yonsei University, Korea
SangKeun Lee	Korea University, Korea
Sang Ho Lee	Soongsil University, Korea
Qing Li	Hong Kong City University, China
Yingshu Li	Georgia State University, USA
Yinsheng Li	Fudan University, China
Xitong Li	MIT, USA
Xue Li	University of Queensland, Australia
Chengfei Liu	Swinburne University of Technology, Australia
Mengchi Liu	Carleton University, Canada
Jiaheng Lu	Renmin University, China
Hua Lu	Aalborg University, Demark
Qiang Ma	Kyoto University, Japan
Shuai Ma	Beihang University, China
Jiangang Ma	University of Adelaide, Australia
Zaki Malik	Wayne State University, USA
Zakaria Maamar	Zayed University, UAE
Hamid Motahari	HP Lab, USA
Weiyi Meng	State University of New York at Binghamton, USA
Azzam Mourad	Lebanese American University, Lebanon
Michael Mrissa	University of Lyon, France
Shinsuke Nakajima	Kyoto Sangyo University, Japan
Miyuki Nakano	University of Tokyo, Japan
Joachim Niehren	INRIA Lille Nord Europe, France
Yuan Ni	IBM Research, China
Werner Nutt	Free University of Bozen-Bolzano, Italy
Satoshi Oyama	Hokkaido University, Japan

Helen Paik	University of New South Wales, Australia
Chaoyi Pang	CSIRO, Australia
Sanghyun Park	Yonsei University, Korea
Zhiyong Peng	Wuhan University, China
Alex Poulovassilis	University of London, UK
Florian Rosenberg	IBM Research, USA
Caspar Ryan	RMIT University, Australia
KeunHo Ryu	Chungbuk National University, Korea
Marc Scholl	Universitaet Konstanz, Germany
Aviv Segev	KAIST, Korea
Mohamed Sharaf	University of Queensland, Australia
Claudia Szabo	University of Adelaide, Australia
Kazutoshi Sumiya	University of Hyogo, Japan
Aixin Sun	Nanyang Technological University, Singapore
Alex Thomo	University of Victoria, Canada
Chaokun Wang	Tsinghua University, China
Daling Wang	Northeastern University, China
Hongzhi Wang	Harbin Institute of Technology, China
Hua Wang	University of Southern Queensland, Australia
Wei Wang	Fudan University, China
Xiaoling Wang	East China Normal University, China
Yinglin Wang	Shanghai Jiaotong Univeristy, China
Jef Wijsen	University of Mons-Hainaut, Belgium
Peter Wood	University of London, UK
Yanbo Wu	University of Adelaide, Australia
Yuqing Wu	Indiana University, USA
Jianliang Xu	Hong Kong Baptist University, China
Linhao Xu	IBM Research, China
Bin Yang	Max Planck Institute, Germany
Xiaochun Yang	Northeastern University, China
Hamdi Yahyaoui	Kuwait University, Kuwait
Jian Yin	Sun Yat-Sen University, China
Muhammad Younas	Oxford Brookes University, UK
Haruo Yokota	Tokyo Institute of Technology, Japan
Jian Yu	Swinburne University of Technology, Australia
Philip Yu	University of Illinois at Chicago, USA
Ming Zhang	Beijing University, China
Rui Zhang	The University of Melbourne, Australia
Ying Zhang	University of New South Wales, Australia
Wenjie Zhang	University of New South Wales, Australia
Xiao Zhang	Renmin University of China, China
Zhenjie Zhang	ADSC, Singapore
Shuigeng Zhou	Fudan University, China
Xiangmin Zhou	CSIRO, Australia
Xuan Zhou	Renmin University, China
Zhangbing Zhou	Institute TELECOM SudParis, France

Table of Contents

Short Papers

Demo Papers

Scalable Integrity-Guaranteed AJAX

Thomas Moyer, Trent Jaeger, and Patrick McDaniel

Systems and Internet Infrastructure Security Laboratory
Pennsylvania State University, University Park, PA 16802, U.S.A.
{tmmoyer,tjaeger,mcdaniel}@cse.psu.edu

Abstract. Interactive web systems are the *de facto* vehicle for implementing sensitive applications, e.g., personal banking, business workflows. Existing web services provide little protection against compromised servers, leaving users to blindly trust that the system is functioning correctly, without being able to verify this trust. Document integrity systems support stronger guarantees by binding a document to the (non-compromised) integrity state of the machine from whence it was received, at the cost of substantially higher latencies. Such latencies render interactive applications unusable. This paper explores cryptographic constructions and systems designs for providing document integrity in AJAX-style interactive web systems. The *Sporf* systems exploits pre-computation to offset runtime costs to support negligible latencies. We detail the design of an Apache-based server supporting content integrity proofs, and perform a detailed empirical study of realistic web workloads. Our evaluation shows that a software-only solution results in latencies of just over 200 milliseconds on a loaded system. An analytical model reveals that with a nominal hardware investment, the latency can be lowered to just over 81 milliseconds, achieving nearly the same throughput as an unmodified system.

1 Introduction

Sensitive, high-value, information–such as banking, enterprise, and intelligence data–are now commonly being distributed through increasingly complex, interactive web systems. Unfortunately, current web systems are not designed to host high-assurance content. At best, the server-side authentication provided by SSL is of limited use, and as it often built on dubious trust relationships [14] and oft-invalid certificates [39]. More fundamentally, web systems provide no content authentication other than identifying the server from which it was obtained. In this current model, there is no way for a user to determine if the content was corrupted by a compromised web server.

Document integrity systems [16,40,22,21,28] augment content with proofs of the correctness of both the document and the system from whence it was received. Such services allow the consumer of the content to validate not only that the document is authentic, but the content was received from an un-compromised system. This prevents otherwise legitimate but compromised systems from providing mis-information, and preemptively prevents that system from silently manipulating and/or exposing user operation and data. For example, a compromised banking site would be immediately detected by the user when attempting to validate the document integrity of the login

screen [34]. The user will simply stop interacting with that site, and therefore no additional damage can be done.

In the Spork project [28], the authors explored the creation of document integrity systems for high-throughput web systems, using the Trusted Platform Module. In order to achieve high throughput, a trade-off for increased latency was made. Such a trade-off poses a challenge for interactive AJAX applications, which require low latency responses to maintain the interactive nature of the web application. What is needed is a document integrity system that supports low-latency responses, to support systems that require low latency, while still sustaining an acceptable throughput.

This paper explores methods and systems designs for providing document integrity in AJAX-style interactive web systems. Chiefly, our *Sporf* system exploits pre-computation to offset runtime costs of providing document integrity. We benchmark a range of *off-line/on-line signature* algorithms and develop new content proof constructions built on them. We detail the design of the Apache-based Sporf server system. A detailed empirical analysis of AJAX applications under realistic workloads is performed. This analysis shows a software-only system results in latencies of approximately 200 milliseconds, with a throughput of 1,500 requests per second. Further modeling shows that a hardware solution, using nominally priced hardware, results in latencies of just over 81 milliseconds, close to that of an unmodified server. In [31], Nielsen states that web application response times lower than 1 second are optimal. Our software-only prototype system can support response times that are approximately 200 milliseconds, as shown in our evaluation. We begin in the next section by providing an overview of document integrity systems and the cryptographic constructions we explore to support low-latency responses.

2   Related Work

In this section we outline several areas of related work. We begin with a description of off-line/on-line signatures. Next we examine mechanisms to provide proofs of system integrity. Finally, we detail mechanisms that provide integrity for web applications.

2.1   Off-line/On-line Signatures

In many operations involving digital signatures, e.g. electronic wallets and high throughput web systems, the signing operation must be very fast. Typical signature schemes, such as RSA [36] and Rabin [33] are too slow for these types of operations. In [15], the authors propose off-line/on-line signature schemes where the heavyweight computations are performed prior to the content being generated, and then a faster signing operation is carried out once the content is presented for signing. This is done by using both *ordinary* public key signature schemes and *one-time* signature schemes, in a two-phase signing operation. In the off-line phase, a one-time key is generated and signed by the ordinary key. This one-time key is then used to sign a single message, or piece of content, in the on-line phase. This on-line signing phase is significantly faster than the ordinary signatures being generated in the first phase. The full construction for off-line/on-line signature schemes can be found in our technical report [29].

2.2   System Integrity Solutions

Clients communicating with web servers over untrusted HTTP connections are given no guarantees about the security of the server or the network communications. Content accessed over SSL, either directly or via a proxy [24], is afforded some protection from network based attacks. However, SSL does nothing to protect the server, or the content hosted on the server. The client cannot be sure that the server, or the content, is not compromised in some way. The SSL certificate simply vouches for the identity of the server, or more specifically of the private key used by the web server. What is needed is a means of providing a "proof" of the server's integrity.

Several proposals exist for software-based attestation, requiring no hardware changes at all. Such solutions are often targeted at mobile devices, but some have also looked at general purpose systems. Early proposals for software-only solutions include those proposed by Spinellis [43] and Kennel and Jamieson [23]. These projects have looked at ways that software can measure itself, in order to provide proof to a remote party that the code executed has not been tampered with. Other projects, such as SWATT [41] and Pioneer [42] look at performing computations over the code being executed that are difficult to compromise, such as executing code that is highly optimized, or walking memory locations in a pseudo-random order. Other works have looked at ways to increase the robustness of such projects [17], or modifying the kernel or shell to measure code before it is executed [27,19]. *Measurement*, in this context, refers to computing a hash of the executable code right before execution. Popular hash functions include MD5 and SHA1.

Software-only solutions have the advantage of not requiring specialized hardware that could be prohibitively expensive. However, such solutions have shown inherent weaknesses, as attacks have been developed on such solutions [49,7]. As such, hardware based solutions are increasing in popularity. Another reason for this increase in popularity is the decrease in the cost of some of the proposed hardware-based solutions.

Hardware based solutions have been proposed as a means of providing tamper-proof storage and execution environments. Projects such as AEGIS [44] and the IBM 4758 [13] provide a secure execution environment. These environments are designed to execute security-sensitive code. One limitation of using hardware in this manner is the cost of deploying the hardware and software[1]. Copilot [32] is another coprocessor based system that monitors the integrity of the kernel. The Trusted Computing Group [46] has developed a set of specifications, including the Trusted Platform Module, TPM, specification [47]. The TPM, unlike other hardware, is designed to be low-cost. Due to the relatively low cost, many commodity systems are now coming equipped with TPMs. Several projects have examined the use of the TPM as a means of measuring the *integrity state* of the system.

Several proposals have looked at using the TPM to provide system integrity reports [25,38,20]. The Linux Integrity Measurement Architecture [38], Linux-IMA or IMA, and its extension, the Policy Reduced Integrity Measurement Architecture [20], PRIMA, measure code before it is loaded and create a hash chain of all executed code.

---

[1] The IBM 4758 and successors are very expensive for consumers to purchase and program, and one is required for every system participating in security-sensitive operations.

IMA measures every single executable and library, while PRIMA uses a policy to determine what code should be measured, reducing the overall size of the measurement list. The measurements are stored in the TPM's PCRs, as described above, and a list of all measurements is stored in kernel memory. When an attestation is requested by a remote verifier, the TPM quote is provided, along with the current measurement list. The verifier can examine the measurement list to determine if the expected software is running on the system and that no un-expected software has run that could potentially compromise the system. One such example of un-expected software would be the Random JavaScript Toolkit [12]. This particular piece of malware is a rootkit that modifies Linux-based Apache webservers. The rootkit contains a small webserver that proxies Apache's responses, by injecting malicious JavaScript before sending the response to the client.

2.3   Web Application Integrity

Several proposals exist looking at providing content integrity for web applications. Some systems look to provide guarantees to the client that the content is correct. SINE [16] and DSSA [40] are two systems that aim to provide such guarantees. SINE provides content integrity to the client, while still allowing the client to retrieve content from caches, instead of requesting content from the server every time. DSSA is a server-side solution that monitors the content hosted by the server. The monitor has a set of known-good pages, and any deviations will cause DSSA to either serve a backup of the content to the client, or simply inform the client that the content is currently unavailable. These solutions still require that the user blindly trust the server, providing no basis for establishing trust that the server is not compromised in any way. Another approach is web tripwires [35] that aims to detect "in-flight" pages changes, by comparing the received content to a known good copy. The tripwire concept assumes that the server is not compromised, again potentially misleading the client.

Other works have looked at utilizing trusted hardware to provide integrity guarantees for the system as well as the content being hosted. Two such systems include the WebALPS project [21,22] and [50]. WebALPS uses the IBM 4758, a secure co-processor developed by IBM to protect the integrity of client-server interactions when the server accesses sensitive client information. In [50], the authors propose a trusted reference monitor, TRM, that protects the integrity and authenticity of peer-to-peer, P2P, systems. The TRM depends on all of the systems in the P2P network to have a TPM, and the clients are required to run secure kernels, such as Microsoft's NGSCB [11]. Such proposals have seen relatively little adoption due to the expensive hardware requirements, or requiring the clients to abandon their current operating systems in favor of new systems.

There has been a large effort to provide secure environments for web applications. Such efforts include [10] and [9]. Both are proposals for new web application programming paradigms. Such solutions work well only if the developer is writing their application from scratch, but does not apply well to existing code bases. Hicks, et. al. [18] looked at ways of building web applications that enforce and end-to-end information flow policy. Other efforts have looked at protecting the integrity of content by specifying the canonical form of the content, such as Document Structure Integrity [30], or

Blueprint [45], or by relying on type systems provided by programming languages, as in [37]. Another approach is to verify computations done on multiple system, as is done in Ripley [48]. While these solutions protect against popular attacks, such as cross-site scripting (XSS), they all assume that the server hosting and generating content is trustworthy with no means of establishing this trust.

3 General Design

In this section we detail the design of a proof construction that uses off-line/on-line signature schemes [15,8] to sign dynamic content, the mechanism used by Sporf. The advantage to this construction is that it removes the TPM from the critical path of binding dynamic content to the system integrity state. We begin with a discussion of document integrity systems and the guarantees they provide.

3.1 Document Integrity Systems

Document integrity systems provide several guarantees about the content they are hosting and the integrity of the system itself. Such systems provide proofs of the origin of the content, as well as proofs of the current system integrity. The following are guarantees provided by document integrity systems:

a) that a document, d, came from a given server, s
b) that the server has a *known integrity state*
c) that the server was in a *known integrity state* at the time the document was generated

Below, we show how such a system can be constructed, from a set of primitives. We leave the details of specific systems for later discussion. We begin by examining the second guarantee, namely the known integrity state. In a system, s, supporting *system integrity proofs*, a verifier connecting to the system will first validate the integrity of the system. This is done using a challenge-response protocol, where the verifier provides a challenge, or nonce, n, and the remote system generates a *system integrity statement*, denoted:

$$\mathsf{IS}_s(n)$$

Here n is a nonce, or challenge, that ensures the freshness of the generated proof. This proof satisfies guarantees b and c. Next, we show how we can build the document integrity proof for a given document.

That document is represented by d, and the server generates a proof for the specific document by computing a cryptographic hash of the document, written $h(d)$. In order to bind a document to the system integrity statement, we *replace the nonce with the document proof*:

$$\mathsf{IS}_s(h(d))$$

This binds the document to the proof, proving to the verifier that the server stored, or generated, document d when the integrity state was reported. To verify, a client validates the $\mathsf{IS}(\cdot)$ and the hash of the received document, $h(d)$. This construction satisfies a and b, but the client can no longer be sure that the proof is relatively fresh. Specifically,

a compromised server can replay such proofs, even after the system has been fully compromised, and the client would be unable to detect an malicious behavior.

To overcome this limitation, the server relies on a trusted time server to provide *verifiable timestamps*, that can be bound to the system integrity statement, in addition to the document proof. Here, | denotes concatenation. The timestamp is written:

$$\mathsf{IS}_{ts}(h(t_i)) \mid t_i$$

Where $\mathsf{IS}(h(t_i))$ is a system integrity proof from the time server, bound to some time, t_i. After obtaining a timestamp from the time server, ts, the server, s, generates the following proof which satisfies the three guarantees outlined above:

$$\mathsf{IS}_s(h(d \mid \mathsf{IS}_{ts}(h(t_i)))) \mid \mathsf{IS}_{ts}(h(t_i)) \mid t_i$$

The hash of the document binds the document to the system integrity proof, the timestamp allows the verifier to determine how fresh the integrity proof is, and the system integrity proof allows the client to validate the integrity of the system. Next, we show how to build a document integrity system that uses commodity trusted hardware and software.

3.2   Example System: Spork

In the Spork project [28], the Trusted Platform Module, TPM [47,46], is used as a means of generating the integrity statements. The TPM provides a limited amount of tamper-evident storage for measurements and cryptographic keys. The measurements are stored in the Platform Configuration Registers, PCRs, and provide a very limited interface for adding values to the set of measurements. The keys stored in the TPM serve a number of different functions, one of which is to sign TPM Quotes. These quotes are the basis for the integrity statements described above. The TPM accepts a nonce and a list of PCRs to report. The TPM reads the selected PCRs and signs the nonce and read values. The quote is written as:

$$\mathsf{Quote}_s(H_s, pcr_s, n)$$

Here, H_s is the key used by the TPM to sign the quote, pcr_s represents the set of reported PCR values and n is the nonce. H_s represents the *identity* of the system, much like an SSL certificate does for secure web transactions. The TPM itself is a passive device, and as such requires support from the system to gather measurements.

Integrity measurement systems gather *measurements* of the current system integrity state. These measurements can later be reported to a verifier to ensure that the system is high integrity. One such system is the Linux Integrity Measurement Architecture [38], which measures executables before they are loaded. A list of these measurements are stored in kernel memory, as well as being reported to the TPM. By including the list of measurements with a TPM quote, a verifier can know what software has been loaded by the system, allowing the verifier to determine if they trust the system, or not. The measurement list is noted as ML_s, where s indicates which system the list is from.

Relying on the TPM leads to very high latencies if each request is signed by the TPM. On average, the TPM takes 900 milliseconds to generate a single quote. In order to amortize this quote generation cost, a Merkle hash tree [26] is used to generate proofs for multiple pieces of content. A client retrieving a proof will get the TPM quote and measurement lists, and a succinct proof from the Merkle hash tree. This allows the

Spork system to sign multiple pieces of content with a single TPM quote, and utilize this same quote to service multiple requests. The root of the hash tree, CPS_r, replaces the single document as the document proof, written:

$$\mathsf{Quote}_s(h(CPS_r \mid \mathsf{Quote}_{ts}(h(t_i)))) \mid \mathsf{Quote}_{ts}(h(t_i)) \mid t_i \mid CPS_r$$

By using the hash tree, the cost of generating a TPM quote is amortized over many documents. This amortization works well for static content, where the TPM can generate a single quote for all content that the web server could potentially serve. However, even using the hash tree, Spork introduces a high amount of latency to dynamic requests, as each client must wait for the TPM to sign a hash tree that includes the requested dynamic content. This additional latency cripples AJAX applications, which require low latency responses to maintain the appearance of desktop-like functionality. Next, we examine the design of a system that reduces the latency for dynamic content.

4   Sporf Overview

Next, we describe the Sporf system, where we examine several potential designs for supporting low-latency, high-throughput integrity-assured web documents, detailing the limitations of each approach. We show how to construct the document proof, before showing the full details of the document integrity system using off-line/on-line signatures to sign dynamic content.

4.1   Binding Using Off-line/On-line Signatures

First, we will introduce some notation that is used throughout the rest of the paper. Keys are denoted as SK and VK for signing and verification, respectively. Keys for one-time signature schemes are super-scripted with ot, i.e. SK^{ot} and VK^{ot}. σ and π represent signatures.

We begin with the system integrity proof, showing how to bind a single document to the system integrity state:

$$\mathsf{Quote}(H_s, pcr_s, h(h(VK) \mid h(VK^{ot}))) \mid VK^{ot} \mid \sigma \mid \pi \mid ML_s$$

where σ is the signature of the one-time key generated with the many-times key, and π is the signature of the document using the one-time key, SK^{ot}. This construction shows that the server, s, with *known integrity state* (guarantee b from Section 3.1), possessed the one-time key-pair used to sign a *document*, d (guarantee a). What is missing is that the document came from the server at the time when the integrity state was reported (guarantee c).Next, we show how to bind multiple verification keys to a single TPM quote and include a recent timestamp from a trusted time server. We rely on cryptographic proof systems, namely a Merkle hash tree, to bind multiple keys to the TPM quote while being able to generate succinct proofs for each key. The leaves of the hash tree are the individual verification keys used to sign documents, and the root of the tree is used as the challenge for the TPM. When a client obtains a document, it obtains a succinct proof for the verification key in addition to the signatures. Figure 1 shows the full construction, using a cryptographic proof system instead of a single key.

$$Q(H_s, pcr_s, h(CPS_r \mid VK \mid Qts)) \mid \underbrace{CPS_r} \mid \underbrace{Q_{ts}} \mid \underbrace{ML_s} \mid \underbrace{VK^{ot} \mid Pf(VK^{ot}) \mid \sigma \mid \pi}$$

server quote	proof sys. root	time server quote	measure- ment list	**document proof**

where:
$$Q_{ts} = Q(H_{ts}, pcr_{ts}, h(t_i)) \mid t_i \mid ML_{ts}$$
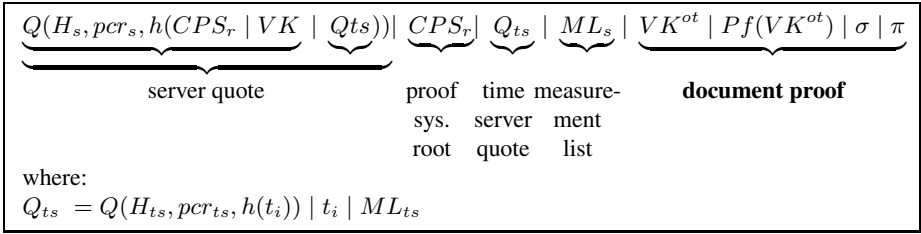
Fig. 1. Quote construction using an off-line/on-line key-pair to sign dynamic content. This construction binds the system's integrity state to a recent timestamp, and recently generated one-time keys. The cryptographic proof system includes all of the static content and recently generated one-time keys. Here $Pf(VK^{ot})$ represents a succinct proof from the hash tree.

The succinct proof is constructed by providing the values of sibling nodes on the path to the root for a given key. Providing this information allows a client to reconstruct the root value from the key it obtains, and compare the computed root to the provided root to ensure that the key is the correct key. This is similar to the method Spork uses to bind multiple documents to a single integrity proof.

4.2   Latency Improvement

The construction in Figure 1 allows the web server to bind a dynamically generated document to the TPM quote, by using the one-time key to sign the dynamic content. This differs from the Spork project which directly binds the content to the proof. In the Spork project, the TPM is on the critical path for serving dynamic content, leading to high latency for each request. With Sporf, the TPM is no longer on the critical path, allowing the system to process requests at much higher speeds, leading to lower latencies for each request.
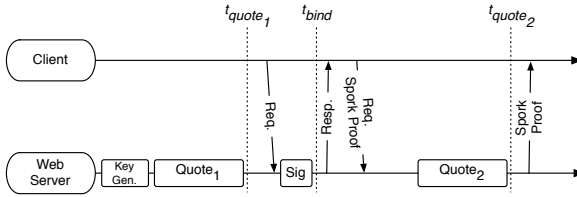
Fig. 2. Timeline for a single request. After the first quote is completed, at time t_{quote_1}, the server is able to sign content. A client makes a request and the server uses a previously generated key to sign the content, time t_{bind} in the figure. At this point, the client has a proof of the server's integrity state up through time t_{quote_1}, and can optimistically begin using the response.

The construction in Figure 1 provides a statement of the system integrity *at the time the keys are generated*, which occurs before the content has been generated. Figure 2 shows the request timeline for a single client requesting content. At time t_{quote_1}, the TPM has generated a quote that can be used to service client requests. When the client

requests content, the server generates a signature for the content, using a key included in the quote. This proof, shown in Figure 1, is generated at time t_{bind}, and sent to the client. The proof in Figure 1, is called a *Sporf-integrity proof*. This distinguishes the proof from a *Spork-integrity proof*, where the client gets the proof at time t_{quote_2}, after the server includes the requested content in a TPM quote.

In order to understand the differences between a Spork-integrity proof and a Sporf-integrity proof, we reconsider each of the guarantees outlined in Section 3.1, in terms of the time at which each guarantee is satisfied. The first guarantee (a) that the document d came from the server. This document comes from the server at time t_{bind}. The second guarantee (b) is that the server, s has a verifiable integrity state. This guarantee is satisfied at time t_{quote_i}, when the TPM generates a quote. The third guarantee is satisfied at time t_{quote_2}, when the client can determine the integrity of the system at time t_{bind}.

This is different than the Spork system, where the binding and reporting occur at the same time, i.e. $t_{bind} = t_{quote_2}$, adding additional latency. For Sporf, the proof is delivered at time t_{bind}, eliminating the latency for obtaining content. While delivering the proof at t_{bind} enables clients to obtain content with lower latency, this delivery also presents a *window of uncertainty*, where the server's integrity cannot be determined by the client.

4.3   Window of Uncertainty and Countermeasures

We define the time between t_{quote_1} and t_{bind}, in Figure 2 as a *window of uncertainty*, as the client cannot be certain of the current state of the system up through time t_{bind}, when the content is generated and signed by the server. In order to validate the integrity of the server during this time, the generated content is included in the *next TPM quote* (Quote$_2$ in Figure 2, i.e. a Spork-integrity proof. The proof obtained is the proof described in Section 3.2, obtained by the client at t_{quote_2} in Figure 2. In this section, we describe mechanisms to mitigate the impact of waiting for the Spork-integrity proof.

To mitigate the impact of waiting for the Spork-integrity proof (at time t_{quote_2}), the client can begin using the data after time t_{bind}, i.e. after the Sporf-integrity proof is obtained, and issue a request for the Spork-integrity proof, to arrive after t_{quote_2}. While waiting for the Spork-integrity proof, any content not validated is highlighted in the client's browser to indicate that it is still not fully validated. Any requests resulting from this content are delayed until the Spork-integrity proof arrives. Next, we describe example applications and how this technique operates.

Example Applications.   Below, we consider two popular applications, and how the developers would integrate Sporf into the application. We first consider the popular Gmail [3] application and also a framework for building AJAX-based instant messaging clients [1]. We show how the typical functions of each application would operate within the Sporf system.

For the first application, consider the web-based email application, Gmail. For this discussion, we will consider what happens when a user receives a message and replies. When the user first logs in, the current contents of the inbox is displayed, and the browser validates the initial requests. The browser will periodically issue AJAX

requests to update the inbox and unread message counts for other labels[2]. When a new message arrives, the browser requests the Sporf-integrity proof as part of the AJAX request. After the initial validation, the view of the inbox is updated. The browser then requests the Spork-integrity proof from the server. Until the Spork-integrity proof is received, the message is highlighted to indicate that the content is not fully validated. The client reads the message while waiting for the Spork-integrity proof and begins writing a reply. If the client finishes the reply and clicks the send button before the Spork-integrity proof is validated, the reply is queued until the Spork-integrity proof is validated, and the client is returned to the inbox. If the proof is valid, the reply is sent in the background, otherwise, the client is notified of the failure and the reply is discarded.

AJAX IM [1] is a framework for building instant messaging clients into web applications. The back-end is a set of PHP scripts, while the front-end is a JavaScript script. Clients send messages to the server, which are then delivered to the other client in real-time. In this case, when the client receives a message from the server, the browser will request the Sporf-integrity proof and highlights the message as only partially validated. The browser requests the Spork-integrity proof after delivering the message. The client can immediately see the message after the Sporf-integrity proof is validated, and can begin writing a response. When the client clicks the send button, the browser first checks that the Spork-integrity proof has been received. If the proof is not received, the message is queued. Once the Spork-integrity proof is validated, the message is sent.

In each of the above examples, the browser is responsible for validating proofs and queueing requests. In future work, we plan to explore the functionality of a Sporf-integrity proof validating browser, and also deploying AJAX applications on Sporf systems.

5   Implementation

This section details the implementation of our Sporf system that supports signing dynamic content using off-line/on-line signature schemes. We begin by describing the various systems, and the functions they perform. In addition to the web server, there is a time server that is generating periodic TPM quotes that use a hash of the current hardware clock value as a nonce, previously described in Section 3.2. The web server includes additional daemons to handle generating TPM quotes, generating off-line/on-line signatures, signing dynamic content, and generating document integrity proofs.

The Sporf daemon is responsible for generating proofs and servicing requests from clients for such proofs. In order to support the off-line/on-line schemes presented in the previous section, the daemon is split into several distinct processes. One process, labeled *Key Generator*, generates off-line/on-line keys, and send these to the main *Sporf daemon* (3 in Figure 3). The Sporf daemon handles signing dynamic content generated by Apache (2 in Figure 3), and storing the signatures of previously generated dynamic content. In addition, the main Sporf daemon carries out several other threads of operation.

---

[2] A Gmail label corresponds to an IMAP folder, except that a message can have more than one label.
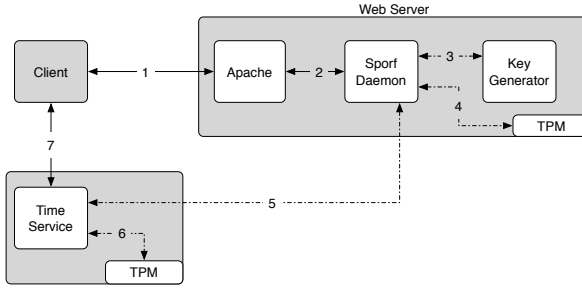
Fig. 3. Detailed system implementation. The numbered arrows represent communications that occur in the Sporf-enabled web server that is serving content integrity proofs. This includes the web server and clients fetching timestamps from the time server, and the daemons generating keys and signing content for the web server.

The main Sporf daemon uses a number of different threads to aid in the content integrity proof generation process. One thread receives off-line/on-line keys from the process generating keys (3 in Figure 3), adding each key to the cryptographic proof system that will be generated in the next TPM quote window. Another thread interfaces with the TPM (4 in Figure 3), to generate the TPM quotes that form the core of the content integrity proofs. This thread also fetches the time attestation from the time server, to be included in the TPM quote generated by the web server (5 in Figure 3). Another thread is responsible for servicing requests for proofs from the web server. This thread compiles all of the proof pieces, such as the content signature, time attestation, TPM quote, and measurement lists, and sends the generated proof back to the server, which returns the proof to the client.

In our current implementation, the main Sporf daemon is started, and spawns the other processes. The number of spawned key generators is configurable, to take advantage of a varying number of processing cores on the web server. This allows the system to be tuned based on expected workloads and available hardware.

6   Evaluation

In this section, we evaluate the throughput and latency of the Sporf system presented in the preceding sections. We begin our evaluation with a comparison to an unmodified Apache web server. These results lead to several optimizations, which are explored to determine the throughput and latency tradeoffs. In addition to the macro-benchmarks, we perform a series of micro-benchmarks to highlight bottlenecks present in the Sporf system.

All tests were performed on Dell PowerEdge M605 blades with 8-core 2.3GHz Dual Quad-core AMD Opteron processors, 16GB RAM, and 2x73GB SAS Drives (RAID 0). Six blades running Ubuntu 10.04.1 LTS Linux kernel version 2.6.32 were connected over a Gigabit Ethernet switch on a quiescent network. One blade ran the Apache web server, one blade ran the time server, and four were used for simulated clients. All

experiments use the Apache 2.2.14 server with mod_python 3.3.1 modules for dynamic content generation. The Spork daemon is written in Python 2.6.5 and uses a custom TPM integration library written in C. All load tests were performed using the Apache JMeter [6] benchmarking tool, version 2.4.

6.1   Microbenchmarks

In the first experiment, we evaluate the throughput of the off-line/on-line signature schemes on our experimental test bed. The implementation of the signature schemes was provided by the authors of [8], and were compiled for the machines in our test environment. Table 1 shows throughput measurements for the off-line/on-line selected schemes from [8]. In this table, we consider only parameter combinations that give the highest throughput for on-line signing. A full table is presented in the Appendix of our technical report [2]. In looking at Table 1, we see that the on-line signing phase for some schemes is able to achieve very high throughputs, specifically, GHR-DL and CS-DL achieving over $50,000$ signing operations per second. This indicates that such a scheme would be ideal for signing dynamic content. For our evaluation, we will consider the schemes that achieve the highest throughputs for on-line signing. This includes the following signature schemes from Table 1: GHR-DL, GHR-DL2, CS-DL, and CS-DL2. These schemes provide the highest on-line signing throughput for a single process, and will introduce the least latency when signing dynamic content.

It should be noted that the schemes labeled GHR-DL and CS-DL provide high throughput in the on-line signing phase. Intuitively, this is due to these signing operations only requiring one integer multiplication operation, unlike other schemes which require more complex operations to complete. Full details are presented in the Appendix of our technical report [2].

Table 1. Benchmarks of off-line/on-line signature schemes. There are a number of parameters that can be tuned, with varying effects on performance. This table only shows the variations that result in the highest throughput for on-line signing. A complete table of microbenchmarks, for various parameter settings, is included in the Appendix of our technical report [2].

Scheme	Off-line Key Gen	Off-line Thpt.	On-line Thpt.	Verify Per Sec.
GHR-OTS	1772.400	525.210	3289.474	762.195
GHR-DL	2348.200	509.165	50000.000	632.911
GHR-RSA	1982.600	510.204	628.141	634.518
GHR-DL2	2182.400	512.295	5813.953	617.284
GHR-RSA2	2001.200	543.478	586.854	672.043
CS-OTS	2282.200	312.500	3289.474	1336.898
CS-DL	2099.600	305.250	62500.000	925.926
CS-RSA	2126.000	304.507	623.441	968.992
CS-DL2	2191.000	309.023	5813.953	961.538
CS-RSA2	1517.200	326.371	585.480	1086.957

Table 2. Macrobenchmarks of the four selected off-line/on-line signature schemes. Jmeter was configured to measure the throughput and latency of the content requests and the proof requests. This configuration allows us to view the individual bottlenecks, as well as the overall throughput and latency experienced by each client. Throughput is measured in requests per second and latency in milliseconds.

	Content		Proof	
	Thpt.	Lat.	Thpt.	Lat.
Baseline	6134.4	80.8	-	-
GHR-DL	384.2	358.9	381.1	316.1
GHR-DL2	390.7	558.6	387.6	256.2
CS-DL	270.8	984.1	266.8	531.7
CS-DL2	274.5	713.6	270.9	415.1

6.2   Baseline Macrobenchmarks

In order to understand the impact of Sporf on serving web documents, we first present a set of macrobenchmarks that examine the throughput and latency characteristics of our Apache web server. In addition to looking at maximum throughput, we examine the latency under several different client populations, ranging from a single client to 512 clients. All tests for maximum throughput will use 512 clients, as adding more clients did not exhibit an increase in throughput, and had an adverse impact on the latency experienced by each client. According to [5], the average size of a page updated via AJAX was approximately 2.5KB, versus 10KB for a full-page refresh. For our experiments, we will use a response size of 2.5KB for each AJAX update.

First, we consider the throughput and latency for an unmodified web server. In our tests, the unmodified Apache web server was able to sustain over 6,100 requests per second, with an average latency of 81 milliseconds per request, as measured by the client. Next, we consider the Sporf-enabled web server. Table 2 shows the throughput and latency average measurements for 512 concurrent clients. The average is taken for a two minute sample, once the system has reached a steady state, i.e. we are ignoring start-up times as these are not indicative of a server's response under load.

In this experiment, we consider the throughput and latency of the Sporf system. For this test, each client requests an update, followed by the proof for that update. Again, we consider the latency experienced by each client as well as sustained throughput.

Table 2 shows the results using the four schemes selected in the previous section. The columns labeled content and proof measure the throughput and latency of each type of request. When looking at the table, it should be noted that the overall throughput for each of the signature schemes is much lower than the throughput shown in Table 1. This is due to the off-line signing phase. There is a single process generating keys and signing them with the off-line key, i.e. running the off-line phase of the signature algorithm. With a single process doing this, the maximum number of keys generated by the system in a single second maxes out at just over 500 keys per second, thus limiting the throughput of the system, and adding additional latency. In later sections, we will consider methods for alleviating this bottleneck.

In addition to measuring the latency under maximum load, we also measured the latency under varying client loads. This reveals how the server responds to various loads,

and what potential bottlenecks exist. We begin with a single client, showing the minimum possible latency, and then consider a maximum of 500 clients. For each client population, clients make serial requests for a period of one minute. All measurements are the average latency during that one minute period, as experienced at the client. Figure 4(a) shows the results for the unmodified web server. We see that the minimum latency experienced is approximately 1.66 msecs, while the average under load is 81.34 msecs. In addition to average latency, the figure shows the median, minimum and maximum latencies for each client population.
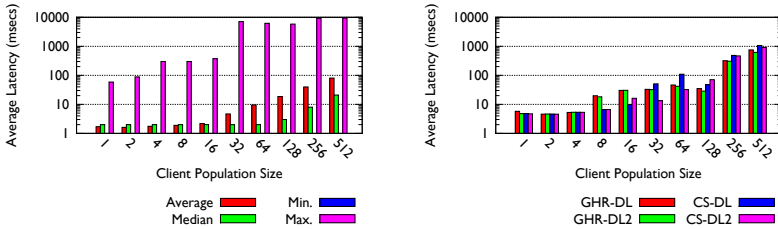
(a) Latency for varying client population sizes for an unmodified Apache web server. The client population size is indicated on the x-axis and the latency is measured in milliseconds.

(b) Average latency of both content and proof as a single transaction. In this case, each set of content–proof requests is grouped, and the average total latency is measured.

Fig. 4.

Figure 4(b) shows the average latency under varying client population sizes for a Sporf-enabled web server. Under heavy client loads, we see the latency increase. The lowest increase in latency is with the GHR-DL scheme, showing an increase from 81.34 msecs to 474.6 msecs. This is due to the clients waiting for fresh keys to be generated, as the number of requests being made per second exceeds the number of keys that can be generated in a single second by Sporf. However, this additional 393 milliseconds is lower than the Spork system, which exhibits an average latency increase of over 1000 milliseconds for dynamic content. In the worst case, i.e. CS-DL and CS-DL2, we see an increase in latency to approximately 1,100 milliseconds.

In looking at Table 2, we see that the latency for content requests is below one second on average, with the additional latency coming from the signing of content. Nielsen states that responses under a second allow the client to perceive little delay [31]. In the next section, we examine an optimization that eliminates the second round-trip to fetch the proof, further reducing the overall latency to just the latency experienced for fetching content.

6.3   Pre-fetching Proofs

A naive solution, outlined above, has each client requesting a proof for each piece of content. This causes each client to make two HTTP requests for every AJAX update. As these updates are happening very frequently, very little is changing in the system's

Table 3. By sending multiple verification keys with a single proof, we eliminate the second round trip to obtain a proof for a large number of content requests. This leads to an increase in the content throughput. Throughput is measured in requests per second and latency in milliseconds.

	10 Keys			25 Keys		
	Content		Proof	Content		Proof
	Thpt.	Lat.	Lat.	Thpt.	Lat.	Lat.
GHR-DL	1511.0	195.9	780.4	1623.0	181.9	588.8
GHR-DL2	1510.3	234.4	787.7	1556.7	188.5	668.8
CS-DL	1476.7	198.9	779.8	1520.1	243.4	617.1
CS-DL2	1466.0	203.3	851.9	1135.4	198.3	669.4

integrity state. We can leverage this by providing a proof to each client that includes verification keys for multiple off-line/on-line signature pairs, along with a single TPM quote. The proof is obtained by the client either with the first request made to the server, or when the current pool of keys is exhausted. When clients request content, the server obtains a key that the client already has an integrity proof for, signs the content and appends the signature to the content. Upon receiving the content and signature, the client has everything needed to validate the integrity proof, without making an additional request to the web server. This completely eliminates the second request, and the additional latency introduced by obtaining the proof after fetching the content.

Table 3 shows the effect of proof pre-fetching for each client. We consider the effect of sending both 10 and 25 keys per integrity proof. In order to better understand the impact of this change and exclude the impact of the off-line signing phase, the system generated a large number of keys, which are then stored, and then signed by the TPM in batches, instead of generating keys in real-time. While this is not how the system will operate in a production deployment, it is useful to understand the potential benefits of Sporf[3]. As shown in the table, it is possible to increase throughput to approximately 1,500 requests per second, while latency for content remains around 200 milliseconds, as compared to approximately 80 milliseconds for serving dynamic content without the content integrity proof. This additional reduction helps to maintain the "sub-one second" goal to maintain the interactivity of the application, as described by Nielsen [31]. The bottleneck in this case is computation, as each client is waiting for a signature of their content.

6.4 Adding Hardware

In supporting large client populations, the system cannot generate keys fast enough to sustain the throughputs of an unmodified web server. In this section, we consider the use of a cryptographic accelerator to support the key generation process. The Silicom PXSC52 [4], which costs just under $500, can sustain a throughput of approximately 17,000 RSA operations per second. Since the off-line phase is based on an RSA signature, if we leverage one of these cards to perform the off-line signing, we can eliminate the bottleneck where clients are waiting for keys to be generated.

---

[3] The off-line/on-line signature implementations are provided by the authors of [8]. We have made no efforts to optimize the implementation of these signature schemes.

To understand the impact adding the accelerator would have on the system, we performed timing tests for the GHR-DL signature scheme. The off-line signing phase can be broken into two steps, with the time for each indicated:

1. Run commitment phase for on-line key (0.292 milliseconds)
2. Sign commitment using many-times key (1.672 milliseconds)

As shown above, the many-times signature operation dominates the off-line signing phase. By moving this signature to the cryptographic accelerator, the off-line signing phase time would drop from 1.964 milliseconds per key to 0.293 milliseconds per key, based on the signature taking 0.001 milliseconds to complete on the accelerator. With this timing, it is possible to generate 3,424 keys per second on a single processor. By adding a second process running the first stage of the off-line signature, we can generate 6,849 keys per second, more than our server's sustained throughput for an un-modified server. This leads to a system where, for a client obtaining an AJAX update, the only latency experienced would be in obtaining a signature for the update, based on the optimizations outlined above. This only adds an additional latency of 0.02 milliseconds per signature, taking the total latency a client can expect down to approximately 81.36 milliseconds, with the server able to sustain the throughputs for an unmodified server, or just over 6,000 requests per second.

7   Conclusion

In this paper, we presented Sporf, a system for generating content integrity proofs for dynamic content. The system provides content integrity proofs for low-latency, high throughput systems, such as AJAX-enabled web applications. We show an in-depth empirical analysis to understand the performance limitations of the prototype Sporf system, and highlight the advantages of leveraging pre-computation. In addition, we explore potential optimizations that allow the system to scale to higher loads without exhausting system resources such as bandwidth. Our results show that a software-only prototype can provide document integrity proofs for dynamic content with approximately 200 milliseconds of latency, with throughputs of 1,500 requests per second. Our analysis shows that a hardware-supported system can provide lower latency, approximately 81 milliseconds, achieving throughputs near that of an unmodified web server.

References

1. Ajax IM – Instant Messaging Framework, `http://ajaxim.com/`
2. Anonymized for submission

3. Gmail, `http://mail.google.com/`
4. PXSC52 - Security Protocol Processor PCI-X Server Adapter / CN1520,
   `http://www.silicom-usa.com/default.asp?contentID=677`
5. Performance Impacts of AJAX Development (October 2010),
   `http://www.webperformanceinc.com/`
   `library/reports/AjaxBandwidth/`
6. Apache: JMeter – Apache JMeter, `http://jakarta.apache.org/jmeter/`
7. Castelluccia, C., Francillon, A., Perito, D., Soriente, C.: On the difficulty of software-based attestation of embedded devices. In: CCS 2009: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 400–409. ACM, New York (2009)
8. Catalano, D., Di Raimondo, M., Fiore, D., Gennaro, R.: Off-Line/On-Line Signatures: Theoretical Aspects and Experimental Results. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 101–120. Springer, Heidelberg (2008)
9. Chong, S., Vikram, K., Myers, A.C.: Sif: enforcing confidentiality and integrity in web applications. In: SS 2007: Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium, pp. 1–16. USENIX Association, Berkeley (2007)
10. Corcoran, B.J., Swamy, N., Hicks, M.: Cross-tier, label-based security enforcement for web applications. In: SIGMOD 2009: Proceedings of the 35th SIGMOD International Conference on Management of Data, pp. 269–282. ACM, New York (2009)
11. Corporation, M.: Microsoft Next-Generation Secure Computing Base,
    `http://www.microsoft.com/resources/ngscb/default.mspx`
12. cPanel: Components of Random JavaScript Toolkit Identified (January 2008),
    `http://blog.cpanel.net/?p=31`
13. Dyer, J.G., Lindemann, M., Perez, R., Sailer, R., van Doorn, L., Smith, S.W., Weingart, S.: Building the IBM 4758 Secure Coprocessor. Computer 34(10), 57–66 (2001)
14. Ellison, C., Schneier, B.: Ten risks of pki: What you're not being told about public key infrastructure. Computer Security Journal 16(1), 1–7 (2000)
15. Even, S., Goldreich, O., Micali, S.: On-line/off-line digital signatures. Journal of Cryptology 9, 35–67 (1996),
    `http://dx.doi.org/10.1007/BF02254791`, doi:10.1007/BF02254791
16. Gaspard, C., Goldberg, S., Itani, W., Bertino, E., Nita-Rotaru, C.: Sine: Cache-friendly integrity for the web. In: 5th IEEE Workshop on Secure Network Protocols, NPSec 2009, pp. 7–12 (2009)
17. Giffin, J.T., Christodorescu, M., Kruger, L.: Strengthening software self-checksumming via self-modifying code. In: ACSAC 2005: Proceedings of the 21st Annual Computer Security Applications Conference, pp. 23–32. IEEE Computer Society, Washington, DC (2005)
18. Hicks, B., Rueda, S., King, D., Moyer, T., Schiffman, J., Sreenivasan, Y., McDaniel, P., Jaeger, T.: An Architecture for Enforcing End-to-End Access Control Over Web Applications. In: Proceedings of the 2010 Symposium on Access Control Models and Technologies, SACMAT 2010 (2010)
19. Iglio, P.: TrustedBox: A Kernel-Level Integrity Checker. In: Proc. of ACSAC 1999, Washington, DC (December 1999)
20. Jaeger, T., Sailer, R., Shankar, U.: PRIMA: Policy-Reduced Integrity Measurement Architecture. In: Proc. of ACM SACMAT 2006 (June 2006)
21. Jiang, S., Smith, S., Minami, K.: Securing web servers against insider attack. In: ACSAC 2001: Proceedings of the 17th Annual Computer Security Applications Conference, p. 265. IEEE Computer Society, Washington, DC (2001)
22. Jiang, S.: WebALPS Implementation and Performance Analysis: Using Trusted Co-servers to Enhance Privacy and Security of Web Interactions. Master's thesis, Dartmouth College (2001)

23. Kennell, R., Jamieson, L.H.: Establishing the genuinity of remote computer systems. In: SSYM 2003: Proceedings of the 12th Conference on USENIX Security Symposium, p. 21. USENIX Association, Berkeley (2003)
24. Lesniewski-Lass, C., Kaashoek, M.F.: SSL splitting: securely serving data from untrusted caches. In: Proc. of USENIX Security Symposium, Washington, DC (August 2003)
25. Loscocco, P.A., Wilson, P.W., Pendergrass, J.A., McDonell, C.D.: Linux kernel integrity measurement using contextual inspection. In: STC 2007: Proceedings of the 2007 ACM Workshop on Scalable Trusted Computing, pp. 21–29. ACM, New York (2007)
26. Merkle, R.: Protocols for public key cryptosystems. In: Proc. of the IEEE Symposium on Research in Security and Privacy, Oakland, CA (April 1980)
27. Mohay, G., Zellers, J.: Kernel and Shell Based Applications Integrity Assurance. In: Proceedings of the 13th Annual Computer Security Applications Conference (ACSAC 1997), San Diego, CA (December 1997)
28. Moyer, T., Butler, K., Schiffman, J., McDaniel, P., Jaeger, T.: Scalable Web Content Attestation. In: ACSAC 2009: Proceedings of the 2009 Annual Computer Security Applications Conference (2009)
29. Moyer, T., McDaniel, P.: Scalable Integrity-Guaranteed AJAX. Tech. Rep. NAS-TR-0149-2011, Network and Security Research Center, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA (March 2011)
30. Nadji, Y., Saxena, P., Song, D.: Document structure integrity: A robust basis for cross-site scripting defense. In: Proceeding of the Network and Distributed System Security Symposium (NDSS 2009) (2009)
31. Nielsen, J.: Designing Web Usability: The Practice of Simplicity. New Riders Publishing, Thousand Oaks (1999)
32. Petroni Jr., N.L., Fraser, T., Molina, J., Arbaugh, W.A.: Copilot–a Coprocessor-based Kernel Runtime Integrity Monitor. In: Proc. of USENIX Security Symposium, San Diego, CA (August 2004)
33. Rabin, M.O.: Digitalized signatures and public-key functions as intractable as factorization. Report TR-212, Lab. for Computer Science, MIT (1979)
34. Raza, M.A.: A Leading Pakistani Bank's Website Got Compromised, http://propakistani.pk/2008/12/26/bank-got-hacked-pakistan/
35. Reis, C., Gribble, S.D., Kohno, T., Weaver, N.C.: Detecting in-flight page changes with web tripwires. In: Proc. of NSDI 2008, pp. 31–44. USENIX Association, Berkeley (2008)
36. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (1978)
37. Robertson, W., Vigna, G.: Static Enforcement of Web Application Integrity Through Strong Typing. In: Proceedings of the USENIX Security Symposium (2009)
38. Sailer, R., Zhang, X., Jaeger, T., van Doorn, L.: Design and Implementation of a TCG-based Integrity Measurement Architecture. In: Proc. of USENIX Security Symposium, San Diego, CA (August 2004)
39. Security Space: Secure Server Survey (June 2009), http://www.securityspace.com/s_survey/sdata/200906/certca.html
40. Sedaghat, S., Pieprzyk, J., Vossough, E.: On-the-fly web content integrity check boosts users' confidence. Commun. ACM 45(11), 33–37 (2002)
41. Seshadri, A., Perrig, A., van Doorn, L., Khosla, P.: Swatt: software-based attestation for embedded devices, pp. 272–282 (May 2004)
42. Seshadri, A., Luk, M., Shi, E., Perrig, A., van Doorn, L., Khosla, P.: Pioneer: Verifying Code Integrity and Enforcing Untampered Code Execution on Legacy Systems. In: Proc. of the 20th ACM Symposium on Operating Systems Principles (SOSP 2005), Brighton, United Kingdom (October 2005)

43. Spinellis, D.: Reflection as a mechanism for software integrity verification. ACM Trans. Inf. Syst. Secur. 3(1), 51–62 (2000)
44. Suh, E., Clarke, D., Gassend, B., van Dijk, M., Devadas, S.: AEGIS: Architectures for Tamper-Evident and Tamper-Resistant Processing. In: Proc. of the 17th International Conference on Supercomputing (June 2003)
45. Ter Louw, M., Venkatakrishnan, V.: Blueprint: Precise Browser-neutral Prevention of Cross-site Scripting Attacks. In: 30th IEEE Symposium on Security and Privacy (2009)
46. Trusted Computing Group: TPM Working Group,
    `https://www.trustedcomputinggroup.org/groups/tpm/`
47. Trusted Computing Group: Trusted Platform Module Specifications,
    `http://www.trustedcomputinggroup.org/developers/`
    `trusted_platform_module/specifications`
48. Vikram, K., Prateek, A., Livshits, B.: Ripley: automatically securing web 2.0 applications through replicated execution. In: CCS 2009: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 173–186. ACM, New York (2009)
49. Wurster, G., Oorschot, P.C.v., Somayaji, A.: A generic attack on checksumming-based software tamper resistance. In: SP 2005: Proceedings of the 2005 IEEE Symposium on Security and Privacy, pp. 127–138. IEEE Computer Society, Washington, DC (2005)
50. Zhang, X., Chen, S., Sandhu, R.: Enhancing data authenticity and integrity in p2p systems. IEEE Internet Computing 9, 18–25 (2005)

Security and Trust in the Web

Vijay Varadharajan

Department of Computing, Macquarie University, Sydney, Australia
`vijay.varadharajan@mq.edu.au`

Abstract. Security and trust issues have been catapulted to the forefront with the dramatic developments in technologies such as web applications, cloud computing, mobile devices and social networking. Though trust has always been a foundational stone of security, the greater dependency of society and economy on information technology have increased the need to consider trust issues more explicitly and systematically. This talk will address some of the key challenges in security and trust in the distributed information infrastructures.

The talk will start with a brief look at some of the recent developments in the threat scenery. Then I will consider the notion of trust in the security world and see how trust issues arise in current ubiquitous computing systems context. Then we will consider a hybrid approach which combines the "hard" attestation based trust with the "soft" social and reputation based trust. Such a hybrid approach can help to improve the detection of malicious entities which in turn can enhance the quality of secure decision making. I will conclude the talk by demonstrating such a trust enhanced security approach using some examples from systems that we have been developing during recent years.

Professor Vijay Varadharajan Bio

Vijay Varadharajan is the Microsoft Chair Professor in Innovation in Computing in Australia at Macquarie University. He is also the Director of Information and Networked System Security Research (INSS) Group at Macquarie University. Before this he was Chair of Computing and Head of School of Computing and IT at University of Western Sydney. Previously, Vijay headed Security Research worldwide for Hewlett-Packard Labs based at European Headquarters at HP Labs Bristol, UK.

Vijay has published over 300 papers in International Journals and Conferences, has co-authored and edited 9 books and holds 3 patents. He has given 20 keynote speeches at international conferences, chaired numerous conferences and has been a program committee member for over 200 conferences all over the world. Vijay has been on the Editorial Board of several journals including the IEEE Transactions in Dependable and Secure Computing, the ACM Transactions on Information Systems Security, Springer International Journal of Information Security and IEEE Security and Privacy. His research work has been supported by industry such as Microsoft, Hewlett-Packard, British Telecom and Fujitsu, as well as government agencies such Australian Research Council (ARC), UK Research Council (EPSRC), Australian Defense (DSD), Dept of Prime Minister and Cabinet Australia and European Union.

Vijay is on the Trustworthy Computing Advisory Board (Microsoft, USA), the SAP (Germany) Security Advisory Board, Australian Government's Peak Security Advisory Group for the Minister of Broadband, Communications and Digital Economy, Australia. He is a member of the Australian Research Council College of Experts in Engineering, Mathematics and Informatics. He has been the Technical Board Director at Australian Computer Society. Vijay is a Fellow of the British Computer Society (FBCS), Fellow of the IEE, UK (FIEE), Fellow of the Institute of Mathematics, UK (FIMA), Fellow of the Australian Institute of Engineers (FIEAust) and Fellow of the Australian Computer Society (FACS).

Estimate Unlabeled-Data-Distribution for Semi-supervised PU Learning

Haoji Hu[1], Chaofeng Sha[2], Xiaoling Wang[1], and Aoying Zhou[1,2]

[1] Shanghai Key Laboratory of Trustworthy Computing, Software Engineering Institute, East China Normal University, China
[2] Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, China

Abstract. Traditional supervised classifiers use only labeled data (features/label pairs) as the training set, while the unlabeled data is used as the testing set. In practice, it is often the case that the labeled data is hard to obtain and the unlabeled data contains the instances that belong to the predefined class beyond the labeled data categories. This problem has been widely studied in recent years and the semi-supervised learning is an efficient solution to learn from positive and unlabeled examples(or PU learning). Among all the semi-supervised PU learning methods, it's hard to choose just one approach to fit all unlabeled data distribution. This paper proposes an automatic KL-divergence based semi-supervised learning method by using unlabeled data distribution knowledge. Meanwhile, a new framework is designed to integrate different semi-supervised PU learning algorithms in order to take advantage of the former methods. The experimental results show that (1)data distribution information is very helpful for the semi-supervised PU learning method; (2)the proposed framework can achieve higher precision when compared with the-state-of-the-art method.

1 Introduction

Classification is a fundamental data mining problem which has been well studied. Traditional classification procedure can be described as follows. Given a set of training data containing category information, after learning a model from the training set, this model can be used to classify the unseen testing instances. There is always an assumption in it that the new instances classified by using this model always belong to one of the categories in the training data.

Unfortunately, this assumption can't be satisfied in many applications. In practice, some of the testing instances may not belong to any of the predefined classes of the original training set, and these instances are always called negative instances. This situation can also be found in our daily life. For example, assuming that there exist two different classes of mails: news and entertainment. If a new email about job description comes, as a result, it's will be assigned to any one of the given two classes. In order to deal with this problem, PU learning is

studied in order to deal with this situation where only the positive and unlabeled instances are given while no negative instance is given.

This paper focuses on the problem of learning from positive and unlabeled examples (or PU learning), where the input instances are multiple(at least 2) labeled positive training classes and unlabeled data. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with small labeled data, to build better classifiers.

Table 1. The comparison of some existing methods

Method	data type	Requirement
s-em[3]	any type	Need more unlabeled negative data
roc-svm[3]	any type	Need more unlabeled negative data
LGN[13]	only text	Need more unlabeled negative data
LiKL[17]	any type	No more constrain on the size of unlabeled negative data

There are some former work on this topic. Table 1 gives the comparison of existing methods. All of these former methods follow an one-vs-all schema. In fact, different algorithms achieve good performance in different unlabeled data distribution. The former methods always perform well, when the unlabeled data contain plenty of negative data. Some methods can only deal with textual data. LiKL need extra attention to tune some parameters. Since the unlabeled data distribution is unknown, it's hard to choose a proper method that best matches the given situation. This paper shows a new idea about integrating different methods behind the intuition of different methods can make up a loss for each other. Our contributions are as follows:

1. An approach to estimate the percentage of negative data in the unlabeled set is first proposed.
2. In order to discover the hidden information in the unlabeled examples, an automatic semi-supervised method based on KL divergence is adopted. This is a general method which is used not just on text but also on other data type, such as category data.
3. With the help of the unlabeled distribution information, a framework that integrates existing approaches to produce a hybrid model is first proposed.
4. A series of experiments are conducted on 20 newsgroups and Reuters corpus, using F-score as measure. Experimental results show that our method can always reach a high precision for both text data and category data.

The rest of this paper is organized as follows: Section 2 presents related works; Section 3 introduces the proposed auto-CiKL approach and describe our way to estimate the proportion of negative data in unlabeled set which is also used to integrate existing algorithms to get a more stable model; Finally, The experimental results are in Section 4, followed by the conclusion in Section 5.

2   Related Work

Several dozen papers have been published on the topic of learning a classifier from only positive and unlabeled training examples. One simple way is to completely ignore the unlabeled examples, namely, learning only from the labeled positive examples, e.g. one-class SVM [1] which aims to approximately covering all labeled positive examples. This work is not always proper. If some reliable negative instances can be identified in the unlabeled data, knowledge provided by unlabeled set will be discarded. And this method makes over-fitting easily.

In semi-supervised learning, unlabeled set is always used for training. These approaches can be covered by two categories. The more common method is (i) to use heuristics to identify unlabeled examples that are likely to be negative,and then (ii) to apply a standard learning method to these examples and the positive examples; steps (i) and (ii) may be iterated. For example, PEBL[2] firstly utilizes 1-DNF[4] to find out quite likely negative examples and secondly employs SVM[5] iteratively for classification. S-EM[3] uses spy technique for extraction of likely negative examples in the first step, and sequently EM algorithm is applied for the parameter estimate. Self-training and co-training also belong to this category. For the self-training method, a classifier is first trained with the small amount of labeled data. The classifier is re-trained and the procedure repeated. Rosenberg et al.[10]apply self-training to object detection systems from images, and show the semi-supervised technique compares favorably with a state-of-the-art detector. In the co-training, Each classifier then classifies the unlabeled data, and teaches the other classifier with the few unlabeled examples (and the predicted labels) they feel most confident. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats. Our work also belongs to this category.

Another common way is to directly modify the training model. Assigning weights somehow to the unlabeled examples is one way. Training a classifier with the unlabeled examples interpreted as weighted negative examples. This approach is used in [6, 7]. B-Pr[8] and W-SVM[9] belong to another probabilistic way. Without needing to extract likely negative examples, they transform the original problem $p(y|x)_{y\in\{+,-\}}$ into a simple sample model $p(t|x)_{t\in\{P,U\}}$, where P is the training set and U is the unlabeled data set.

It has been noticed that constraining the class proportions on unlabeled data can be important for semi-supervised learning. Without any constrains on class proportion, various semi-supervised learning algorithms tend to produce unbalanced output. For this reason, various semi-supervised learning methods have been using some form of class proportion constraints. For example, Zhu et al.[11] use a heuristic class mean normalization procedure to move towards the desired class proportions; S3VM[18] methods explicitly fit the desired class proportions. However, in these methods the class proportion constraint is combined with other model assumptions. To our best knowledge, this paper first introduce an unlabeled data distribution estimate method for PU problem.

3   The Proposed Algorithm

3.1   Preliminaries

The KL Divergence[16] between the probability distribution $P = \{p_1, \ldots, p_n\}$ and $Q = \{q_1, \ldots, q_n\}$ is defined as:

$$KL(P \parallel Q) = \sum_{i=1}^{n} p_i \lg \frac{p_i}{q_i}$$

In this paper, the KL divergence is calculated just as [17] between the posterior probabilities of every instance d_i in the unlabeled set and the prior probabilities of each class, and then the revised formula of KL divergence is defined as:

$$KL(d_i) = \sum_{j=1}^{|C|} p(c_j|d_i) \lg \frac{p(c_j|d_i)}{p(c_j)}$$

where $p(c_j|d_i)$ is the class posterior probability of instance d_i belonging to the j-th class c_j; $p(c_j)$ is the prior probability of each class; $|C|$ is the number of known or predefined class labels appearing in the training set. The intuition of this revision is that, if an instance belong to a labeled category the KL divergence from posterior probability distribution to prior probability distribution is large; if an instance is negative it will belong to any labeled category equally that lead to small KL Divergence.

For the unlabeled set(labeled as U), the KL divergence of every instance is calculated, and the instances

$$d_i = \underset{d_i \in \{d_t \mid \arg\max_{c_k \in C} (p(c_k|d_t))\}}{\arg\max} (KL(d_i))$$

and

$$d_j = \underset{d_j \in U}{\arg\min} (KL(d_j))$$

are obtained as the most likely corresponding sub-positive(c_k) and negative examples respectively.

3.2   Problem Description

Given that a training set(P) only containing positive examples, from multiple (at least 2) classes, without negative ones, and one unlabeled set(U) containing both positive and negative examples, our task is to construct a classifier(C), finding all likely negative examples(U_n) hidden in the unlabeled set and it is formulated as follows: input$(P, U) \overset{C}{\Rightarrow}$ output(U_n).

3.3   Auto CiKL Approach

This work is based on our former work LiKL[17], which contains three steps. In the first two a probablistic classifier C_1 is trained from the data of positive set. Then use the posterior probability returned by C_1 to calculate the KL divergence of every instance in the unlabeled set and pick up reliable positive instances and reliable negative instances. In the last step, with the help of instances identified from the unlabeled set, the final classifier to identify the instances that don't belong to any class in the training set is trained.

In this paper, a improvement for LiKL called Auto CiKL is introduced. The difference between Auto CiKL and LiKL lies in the first two steps.

The first step is to find the most likely positive examples. The labeled set P is used to learn a classifier for getting posterior probability. Through the posterior probability, the KL divergence of each instance in the unlabeled set can be calculated out. Pick out the top K largest and add them into labeled set with labeling them to corresponding class. If the percentage of positive instances in the unlabel data is unknown, it's hard to pick up a proper K, which is the shortcoming of LiKL. In Auto CiKL we use a function $EstimateProportion(P,U)$ to estimate the proportion of negative instances. Then we can easily get the proportion of positive instances. After getting the proportion, we can decide the K based on it. The function $EstimateProportion(P,U)$ will be given in the next subsection.

In the second step, a plenty of labeled data are available. The new training set $(P+S_p)$ is used to build a new classifier FN to classify the remaining unlabeled set $(U-S_p)$, and find the top N most likely negative instances. All the negative instances are denoted as S_n;

Now a training set $(P+S_p+S_n)$ is obtained which contains both positive instances and negative instances. Use $(P+S_p+S_n)$ to train the final classifier model and classify the instances in the set $(U-S_p-S_n)$. Some classification algorithms such as SVM and logistic regression can be used in the last step.

3.4   Distribution Estimation for Unlabeled Data

The step 1 and step 2 of Auto CiKL suffer from the problem of setting the value of K and N respectively. It's difficult to set these two parameters if any extra information is unknown. The reason is as follows. If a small value is given to K, when the percentage of positive instances in the unlabeled set is large, and too many positive instances will be left in the unlabeled set so that the classifier which use the remaining unlabeled set as negative instances set will be ineffective. On the contrary, if a large value is given to K, when the percentage of positive instances in the unlabeled set is small, too many negative instances will be introduced into the labeled set as positive instances, that will pollute the labeled training set. The similar analysis can be found for the value of N.

It's obvious to give different values for different density of positive or negative instances in the unlabeled set. In this paper, we use the density of unlabeled set stands for the percentage of negative instances in the unlabeled set.

(a) Low Density   (b) High Density
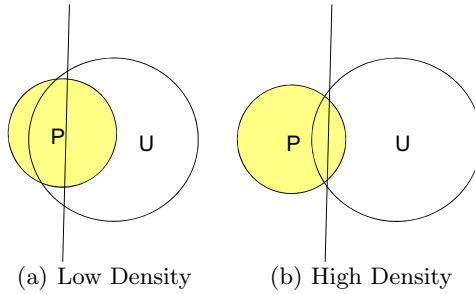
Fig. 1. Classifiers for Labeled Set and Unlabeled Set of Different Density

Unfortunately, the density of unlabeled set is unknown. That means it is important to estimate the density of the unlabeled data.

The intuition of our method is presented in the Figure 1. The P set stands for the labeled set, while the U set represents the the unlabeled set. Label the instances in the P as the positive ones, and label the instances in the U as the negative one. Then use all of them to learn a classifier. (a) and (b) show the situation that the U set contains low percentage and high percentage of negative instances respectively. It's obviously that the classifier in the (b) can get a higher recall for the positive instances than that in the (a). The reason is that, the more real negative instances which the unlabeled set contains, the easier classifier separate the P and the U properly.

The detailed algorithms are shown in Algorithm 1 and 2.

In the Algorithm 1, Num instances are generated for different proportions. The more instances we generate, the higher accuracy for estimation. The main idea of the *generate_instances* method is to find the distribution of each attribute for each category and to generate positive instances based on these distributions of attributes in the training set P. Each attribute of each positive instance in UT is generated one by one through the distribution found in P.

In the Algorithm 2, after the training set is produced, a model used for estimating proportion could be learned. Take the recall of the P as the input of the model. The estimation for the negative instances proportion of unlabeled data is outputted from this model. As the estimate proportion of negative instances is available, proper values can be chosen for K and N respectively.

3.5   Integrating Different Methods in the New Framework

From the experiment results showed in Figure 6, we can conclude that no one method can achieve the best performance in the all cases. Different methods have their different best fit situations. If the knowledge about the proportion can be accessed, choosing a proper approach for different situation is possible. In another words, these different methods could be integrated into a mixture method. This paper uses the proportion knowledge to integrate KL method and ROC-EM method to achieve a better mixture model, which can always stay good performance than just using only one of them.

Algorithm 1: GenerateTrainingSet(ProList, Num, P, U)

> Input: list ProList contains different proportions,
> the number of instances for each proportion will be
> generated Num, training set P, unlabeled set U;
> Output: training set T used for learning estimator
> for proportion;

1 **for** each proportion p_i in ProList **do**
2   **for** j from 0 to Num-1 **do**
3   PT ← P;
4   UT ← $generate\_instances(|U| \cdot (1 - p_i)$,P);
5   UT ← $random\_select(|U| \cdot p_i, D)$;
6   train the classifier C to classify PT and UT;
7   get the recall r_{ij} using C to classify PT;
8          $TrainingSet \leftarrow (r_{ij}, p_i)$;
9   **end**
10 **end**
11 **return** $TrainingSet$;

Fig. 2. Generating training set for estimate proportion

Algorithm 2: EstimateProportion(P, U)

> Input: training set P, unlabeled set U;
> Output: likely negative instances proportion of U

1 T = GenerateTrainingSet(ProList, Num, P, U);
2 model $m = logistic\_regression(T)$;
3 label instances in P as positive;
4 label instances in U as negative;
5 find classifier C which separates P
   from U;
6 calculate the recall r for instances in P using C;
7 Using m to estimate the negative instances proportion
   p of U with the recall r;
8 **return** p;

Fig. 3. Estimating the negative instances proportion of unlabeled set

Based on the estimate of unlabeled data distribution, a new framework for semi-supervise learning can be proposed. This is based on the estimate of unlabeled data distribution. With the knowledge of distribution, a framework that integrates different performance methods is given. This framework can integrate some former methods which have different performance in different unlabeled data distribution. It automatically choose the proper approach according to the distribution knowledge.The experimental results demonstrate that that hybrid method can achieve a globally high performance.

Algorithm 3: Integrate(*methods_list*, *proportion_list*)

   Input: a list contains all the approaches that will be
integrated, a list contains different proportions;
   Output: a hybrid approach;
1 **for** each proportion p_i in *proportion_list*
2   choose the method m_j in *methods_list*
3   that has the best performance at proportion p_i case;
4 **end**
5 **return** *hybridapproach*;

Fig. 4. New framework for semi-supervise learning

4   Experimental Evaluation

The objective of this section is to evaluate our proposed approach in terms of
learning accuracy. The experiments are conducted on Intel 2.0 GHZ PC with
2 GB of RAM. The The classisifers used in our approach are implemented in
Weka[1] environment. And the existing PU learning methods such as roc-em are
downloaded from http://www.cs.uic.edu/∼liub/LPU/LPU-download.html.

4.1   Data Sets

We used the benchmark 20 Newsgroup collection[19] and Reuters corpus[20].

   20 Newsgroup has approximately 20000 documents, divided into 20 differ-
ent small subgroups respectively, each of which corresponds to a different topic.
Firstly, four subgroups are chosen from computer topic and two topics from sci-
ence respectively, i.e. comp.graphics, comp.ibm.hardware, comp.mac.hardware,
comp.windows.xsci.crypt, sci.space, $C_4^1 C_2^1 = 8$ pairs of different experiments.
For each pair of classes, i.e. selecting one class from {graphics, ibm.hardware,
mac.hardware, windows.x}×{crypt, space} respectively as two positive classes,
e.g. **graphics×crypt**, an equal part of documents are chosen randomly for train-
ing as corresponding positive instances, and the rest as unlabeled positive data
in unlabeled set; Then some examples are extracted randomly from the rest
18 subgroups are viewed as unlabeled negative examples in unlabeled set, and
the number is $\alpha \times |U|$, where α is a proportion parameter, showing the per-
centage of negative examples in the unlabeled set, and $|U|$ is the number of all
instances in the unlabeled set. The similar operation is done to Retuers, and
the topic combinations {acq-crude, acq-earn, crude-interest, crude-earn, earn-
interest, interest-acq} are chosen as the positive topic.

Feature Extraction. All the documents in our experiments are modeled by
Vector Space Model. Feature selection is very important in textual classification.
Instead of using PCA or LDA for dimensionality reduction, a light but effective

---

[1] http://www.cs.waikato.ac.nz/ml/weka

way which is borrowed from information retrieval is used. TF-IDF is a measure used to reflect the importance of the words in documents. The words with high TF-IDF value for each class are retained. The intuition of this method is that it's common to see some certain words in some certain topics. This is similar to the fundamental of LDA. And the result is readable.

4.2   Experimental Results

We perform 10 times hold-out tests with random data partitions to get the average F-score value as the final result. In the experiments, α is the ratio of the unlabeled negative examples compared to the unlabeled set. E.g. $\alpha = 0.1$ means that the number of the unlabeled negative examples is only 10% of the unlabeled set.

Performance for Auto-KL Method
Figure 5 records the comparison of the performance between using proportion estimate and giving the real proportion. KL-Auto represents the former and KL-ByHand represents the latter respectively. Both of them have the trend that the F scores increase with the proportion of the negative instances increase. KL-Random means randomly giving values to K and N. It's obvious that the performance of KL-Auto decreases because of the mistakes made by estimator. But the result is much better than the KL-Random, in which all the N and K are chosen randomly. That means our proportion estimate approach is so effective that the estimate given by our method is close to the real proportion.

Performance for Different Algorithms
Figure 6 records the F scores of Spy-SVM, ROC-EM and KL-Auto. The ROC-EM means using rocchio technique for the first step to identify reliable negative instances and using EM for the second step to build final classifier. The Spy-SVM means use the spy technique for the first step and the use SVM step for the second step.

   As shown in Figure 6, no one approach can achieve the best performance all the time. It's easy to see the KL-Auto outperforms the other two methods dramatically in the case of 0.1 and 0.2. KL-Auto perform quite well in the other cases, but not as well as these method proposed by Liu. In the contrary, ROC-EM can get high F score in the high proportion case, but have bad performance in the case of 0.1 and 0.2. Each approach has its best fit situation. In another words, if the proportion of the unlabeled data set is unknown, when only one of these methods is chosen, result may fall into the bad case of these methods.

Performance for Integration of Different Approaches
Since the proportion of negative instances in unlabeled data can be estimated, not just one approach can be used. Methods that can make up a loss for each other can be picked up to produce a hybrid approach. Figure 7 shows the result of integration of spy-svm and KL-Auto. It's clearly to see that the hybrid approach can almost achieve the best all the time.

(a) 20 Newsgroup   (b) Reuters
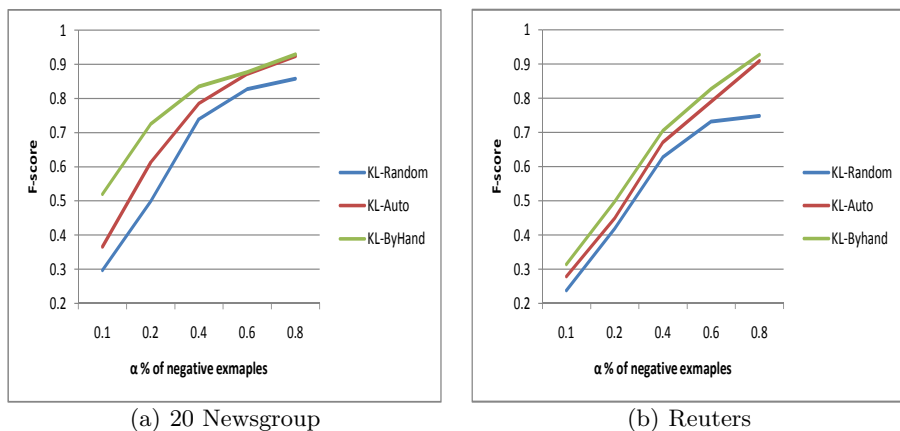
Fig. 5. Comparison between given estimate proportion and given real proportion

(a) 20 Newsgroup   (b) Reuters
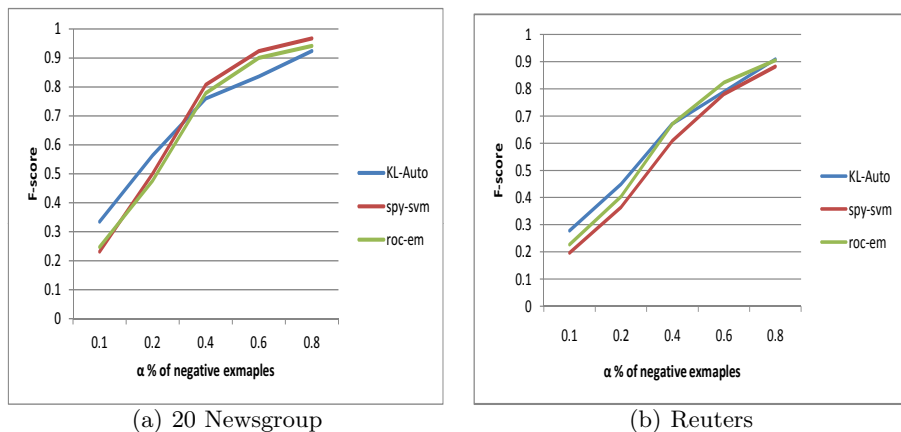
Fig. 6. The F scores of different algorithms in different negative instances proportion

From the above experiments, we can draw the following conclusion.

(1) Using the estimate for proportion, the parameters can be automatically chosen for the KL method effectively. The KL method outperform than existing methods in the low proportion case.

(2) No one algorithm can always achieve good performance, every approach has its best fit proportion.

(3) The new framework for semi-supervise learning can always generate a global optimum hybrid method.

(a) 20 Newsgroup

(b) Reuters
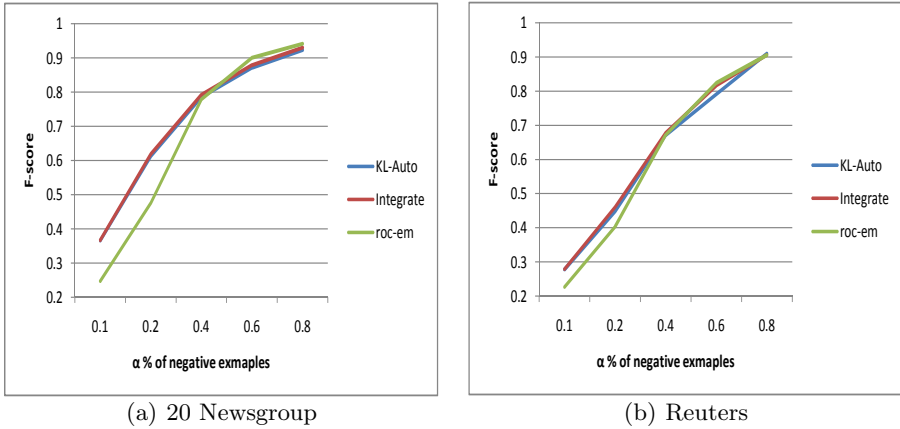
Fig. 7. Performance for the hybrid method

5   Conclusion

In this paper, the problem of learning from positive and unlabeled examples is tackled by using a novel approach called Auto CiKL and propose a framework that can integrate other algorithms to obtain a global high performance hybrid method. The CiKL method can achieve good performance when the negative instances proportion is low. And the hybrid method can always perform better. In the further work, we will further study how to improve the precision of our estimate method and give theoretical analysis for our approach.

References

1. Manevitz, L.M., Yousef, M., Cristianini, N., Shawe-taylor, J., Williamson, B.: One class svms for document classification. Journal of Machine Learning Research 2, 139–154 (2001)
2. Yu, H., Han, J., Chang, K.C.C.: Pebl: Positive example based learning for web page classification using svm. In: KDD (2002)
3. Li, X., Liu, B., Dai, Y., Lee, W.S., Yu, P.S.: Building Text Classifiers Using Positive and Unlabeled Examples. In: ICDM (2003)
4. Denis, F.: PAC Learning from Positive Statistical Queries. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) ALT 1998. LNCS (LNAI), vol. 1501, pp. 112–126. Springer, Heidelberg (1998)
5. Cortes, C., Vapnik, V.: Support vector networks. Machine Learning 20, 273–297 (1995)

6. Lee, W.S., Liu, B.: Learning with positive and unlabeled examples using weighted logistic regression. In: Proceedings of the Twentieth International Conference on Machine Learning (2003)
7. Liu, Z., Shi, W., Li, D., Qin, Q.: Partially Supervised Classification – Based on Weighted Unlabeled Samples Support Vector Machine. In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS (LNAI), vol. 3584, pp. 118–129. Springer, Heidelberg (2005)
8. Zhang, D., Lee, W.S.: A simple probabilistic approach to learning from positive and unlabeled examples. In: UKCI (2005)
9. Elkan, C., Noto, K.: Learing classifiers from only positive and unlabeled data. In: KDD (2008)
10. Rosenberg, C., Hebert, M., Schneiderman, H.: Semi-supervised selftraining of object detection models. In: Seventh IEEE Workshop on Applications of Computer Vision (2005)
11. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: The 20th International Conference on Machine Learning (2003a)
12. Vapnik, V.: Statistical learning theory. Wiley-Interscience (1998)
13. Li, X.L., Liu, B., Ng, S.K.: Learning to Identify Unexpected Instances in the Test Set. In: AAAI (2007)
14. Zhu, X.J.: Semi-Supervised Learning Literature Survey. Technical Report 1530, Dept. Comp. Sci., Univ. Wisconsin-Madison (2006)
15. Wang, X., Xu, Z., Sha, C., Ester, M., Zhou, A.: Semi-supervised Learning from Only Positive and Unlabeled Data Using Entropy. In: Chen, L., Tang, C., Yang, J., Gao, Y. (eds.) WAIM 2010. LNCS, vol. 6184, pp. 668–679. Springer, Heidelberg (2010)
16. Cover, T., Thomas, J.: Elements of Information Theory. Wiley Interscience, Hoboken (1991)
17. Xu, Z., Sha, C.F., Wang, X.L., Zhou, A.Y.: Semi-supervised Classification Based on KL Divergence. Journal of Computer Research and Development 1, 81–87 (2010)
18. Bennett, K., Demiriz, A.: Semi-supervised support vector machines. In: NIPS 11, pp. 368–374 (1999)
19. http://people.csail.mit.edu/jrennie/20Newsgroups
20. http://www.daviddlewis.com/resources/testcollections/reuters21578/

Disputed Sentence Suggestion
towards Credibility-Oriented Web Search

Yusuke Yamamoto

Kyoto Univeristy, Yoshida-honmachi, Kyoto, Japan
yamamoto@dl.kuis.kyoto-u.ac.jp

Abstract. We propose a novel type of query suggestion to support credibility-oriented Web search. When users issue queries to search for Web pages, our system collects disputed sentences about queries from the Web. Then, the system measures how typical and relevant each of the collected disputed sentences are to the given queries. Finally, the system suggests some of the most typical and relevant disputed sentences to the users. Conventional query suggestion techniques focus only on making it easy for users to search for Web pages matching their intent. Therefore, when users search for Web pages to check the credibility of specific opinions or statements, queries suggested by conventional techniques are not always useful in searching for evidence for credibility judgments. In addition, if users are not careful about the credibility of information in the Web search process, it is difficult to be aware of the existence of suspicious Web information through conventional query suggestions. Our disputed sentence suggestion enhances users' awareness of suspicious statements so that they can search for Web pages with careful attention to them.

1 Introduction

Nowadays, the Web is a huge information resource. People can easily and freely obtain information through Web search engines and specific Web sites. However, the problem of Web information credibility is emerging [1,2].

The quality control of Web information is generally insufficient due to low publishing barriers. As a result, there is a large amount of incorrect and unreliable information on the Web that can have detrimental effects on users. For instance, Sillence et al. reported that there are more than 20,000 medical Web sites on the Web, but over half of them have not been checked by medical experts [1]. On the other hand, Nakaura et al. reported that a lot of Web users trust Web information to some extent [2]. In particular, for Web search engines, they also reported that more than 50% of search engine users perceive the search results to be somewhat credible.

Although Web information credibility is becoming more critical, conventional Web search engines still focus on searching for only popular and relevant information [3,4]. In addition, they provide few clues to judge information credibility from Web search results. For instance, suppose that a user is searching for Web pages about *effective diet*. A lot of Web pages describe various diet methods. However, as in the case of the Atkins diet[1], Web pages containing suspicious information are often listed with

---

[1] Atkins diet: http://en.wikipedia.org/wiki/Atkins_diet

(a) Disputed sentence suggestion   (b) Conventional keyword suggestion
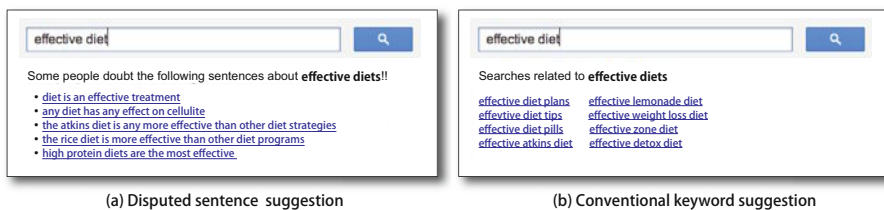
Fig. 1. (a) Disputed sentence suggestion and (b) conventional keyword suggestion

high rankings as popular and relevant Web search results, although there exists counter evidence against the information. Conventional Web search results provide only titles, a snippet, and URLs. This makes it difficult for users to check the credibility of such Web pages and obtain credible Web information using the conventional Web search engines. In worst cases, if users are not skeptical about Web pages, they can be mislead by incorrect information without knowing it. Therefore, automatic tools for alerting users and helping them judge Web information credibility are becoming increasingly necessary.

In this paper, to support users' credibility judgment in the Web search process, we propose a novel type of query suggestion to make users aware of suspicious Web information. When users issue queries to search for Web pages, our proposed system suggests some disputed sentences about the given queries. Fig.1-(a) and (b) show the case where a user issues keyword query *effective diet* to our Web search engine and to a conventional Web search engine, respectively. For this query, the conventional Web search engine suggested keywords *effective diet plans*, *effective atkins diet*, and so on, in Fig.1(b). As this example shows, conventional query suggestion techniques focus on supporting users in making their intent clearer and making it easy to search for Web pages matching their intent. In addition, suggested queries are often keywords. On the other hand, our implementation suggested that some phrases such as *"diet is an effective treatment"* and *"the atkins diet is not any more effective than other diet strategies"*, were disputed by some Web pages, in Fig.1(a). We think that disputed sentence suggestion has the following advantages over conventional keyword query suggestion techniques for supporting users' credibility judgment in the Web search process:

- Users can clearly recognize the existence of suspicious statements on the Web even if they are not overly concerned about the credibility of Web information.
- Users can find some queries to collect clues for credibility judgment of suspicious information from the viewpoint of counter evidence.
- Users can intuitively understand the meaning of disputed statements as opposed to just reading a list of keywords.

The remainder of the paper is organized as follows. In the next section, we discuss related work. Section 3 describes our methods for collecting disputed sentences from the Web and ranking them. In Sections 4, we show the effectiveness of our system for supporting users' credibility judgment in the Web search process, comparing it to some query suggestion techniques. The last section concludes the paper and outlines our future research directions.

2 Related Work

2.1 Query Suggestion

There has been a lot of work on query suggestion, including query expansion [5] and query substitution [6,7]. Cui et al. studied a method to expand given queries by considering the gap between document space and query space [5]. Boldi et al. proposed a method to classify users' query reformulation operations into *generalization*, *specialization*, *error correction*, or *parallel move* towards query substitution [6]. Kotov et al. addressed a framework to automatically generate potentially interesting questions from Web pages and suggest them in the Web search process, so that Web search engines can naturally lead the users to Web search results they want [7]. These approaches are very sophisticated. However, they still focus on only mismatching problems between users' intent and given queries to improve users' Web search experience. Even if some queries are suggested, it is still difficult to call users' attention to suspicious Web information or to support users in collecting clues for credibility judgment from the Web.

2.2 Measuring Web Information Credibility

Some researchers have developed methods to measure the credibility of specific Web contents from specific credibility aspects. Especially, research aimed at measuring the credibility of contents on social networks like Yahoo Answers[2] and Twitter[3] is becoming more popular. Suryanto et al. focused on the expertise of answerers on Q&A sites to evaluate answers posted on them [8]. Castillo et al. proposed a method to automatically judge the credibility level of news propagated through Twitter by analyzing tweets and re-postings about news [9]. To measure the credibility of Web pages and Web search results from various viewpoints, some researchers have developed prototype systems [10,11]. These systems visualize scores of Web search results on the basis of important credibility criteria, such as popularity, content typicality, and content freshness. If users want to briefly filter out non-credible Web information, measuring the credibility of Web information can be useful.

2.3 Supporting Users' Credibility Judgment on the Web

Some studies have focused on promoting and helping users judge credibility by themselves. Pirolli et al. used WIKIDASHBOARD, the system for visualizing edit histories of Wikipedia articles, to study how the system affects users' credibility judgments on those articles [12]. Ennals et al. developed DISPUTE FINDER, a system that highlights suspicious sentences on Web pages users are browsing [13]. DISPUTEFINDER can highlight only suspicious sentences in its database and it is not robust in a Web search process where different Web search results are shown for different queries. On the other hand, our system is robust. In addition, our system enables users to get an overview of suspicious statements for given queries. Some researchers have developed methods to find alternative statements for checking the credibility of suspicious statements [14,15].

---

[2] Yahoo Answers: http://answers.yahoo.com/
[3] Twitter: http://twitter.com/
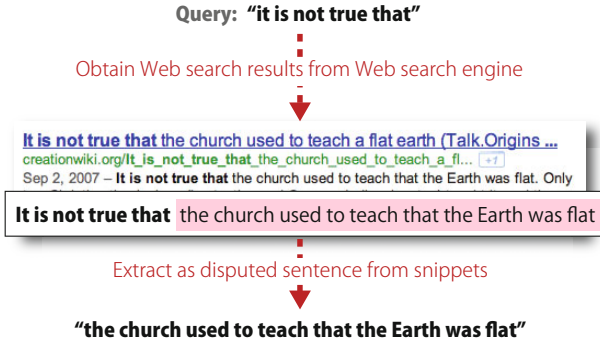
Fig. 2. Example of disputed sentence extraction from snippets of Web search results

Table 1. Linguistic patterns to collect disputed sentences from the Web

Linguistic pattern
"no proof that", *"mistakenly believe that"*, *"no evidence that"*, *"lie that"*
"it is not true that", *"it is not clear that"*, *"it is unlikely that"*, *"it is wrong that"*
"it is not to say that", *"it is denied that"*, *"into believing that"*
"misconception that", *"myth that"*, *"it is speculated that"*, *"it is delusion that"*

3   Disputed Sentence Suggestion

In this section, we describe our method for searching for disputed sentences about a given query on the Web and suggesting disputed sentences to users. Our system works as follows:

1. Collect disputed sentence candidates about a given query from the Web.
2. Score the disputed sentence candidates by considering the typicality of the disputed sentences and their relevancy to the given query.
3. Suggest top-k of the disputed sentences to users

3.1   Extracting Disputed Sentences from the Web

Detecting sentences that other documents dispute is quite hard because it requires deep natural language processing for large corpora. To bypass this problem, Ennals et al. proposed a method to collect disputed sentences from the Web by using linguistic patterns [16]. We apply their method to collect disputed sentences about a given query.

Ennals et al. prepared a set of dispute clue patterns that could indicate that a statement S is disputed, such as *"the misconception that S"* and *"it is not true that S"*. Then, as shown in Fig.2, they issued the prepared linguistic patterns as phrase queries into Web search engines. After that, they extracted the sentences that appeared just behind one of the patterns from obtained Web search results.

While Ennals et al. focus on collecting a large indefinite number of disputed sentences in batch processing, our goal is to collect disputed sentences related to a user's

query in real time. To achieve this goal, we selected 15 of the 54 effective linguistic patterns that Ennals et al. suggested in their paper. The linguistic patterns are shown in Table 1. We implemented the method to collect disputed sentences by using these selected linguistic patterns. Our method works as follows:

1. Issue a given query and each of the prepared linguistic patterns into Yahoo Search Web API[4] (e.g. given keyword query *global warming cause*, the system issues expanded query *global warming cause AND "it is not true that"*).
2. Split snippets of Web search results into sentences using some delimiters (e.g. ".", ",", !, ?, *and*, *but*, and so on).
3. Extract sentences that appear just behind one of the prepared linguistic patterns from collected sentences as disputed sentences.
4. If the extracted sentences start with a specific stopword, such as a pronoun, a conjunction, a preposition, or a meaningless term, the system eliminates them from a list of the disputed sentences.

3.2 Ranking Disputed Sentences

After collecting disputed sentence candidates from Web search results about a given query, the system ranks the candidates to suggest useful disputed sentences to users.

Here we denote a given query and a disputed sentence by q and d, respectively. We estimate the importance of each collected disputed sentence $D = \{d_1, d_2, \ldots, d_n\}$, focusing on the relevance to given query q. Then we order them by using the following equations on the query likelihood model [3]:

$$p(d|q) \propto p(d) \prod_{t \in q} ((1 - \lambda)p(t|M_c) + \lambda p(t|M_d)) \tag{1}$$

$$p(q|M_d) = \prod_{t \in q} \frac{tf_{t,d}}{L_d} \tag{2}$$

where M_d is the language model of disputed sentence d, $tf_{t,d}$ is the term frequency of term t in disputed sentence d, and L_d is the number of terms in disputed sentence d. λ is a weighting parameter and $0 < \lambda < 1$. M_c is the language model built from the entire disputed sentence collection D.

When using the query likelihood model, the prior probability of a document $p(d)$ is often treated as uniform across all d. However, we calculate $p(d)$ to consider how popular the collected disputed sentences are on the Web.

One way to determine the popularity of disputed sentences on the Web is to count how many times they appear. However, as exemplified by sentences *"a mobile phone is bad for your health"* and *"mobile phones are a health risk"*, the same statements are often expressed in different words. Therefore, it is difficult to measure the popularity or typicality of sentences by simply using their frequency. To solve this problem, we use the *LexRank* algorithm [17]. In this algorithm, a graph is created from

---

[4] Yahoo! Search Web API:
  http://developer.yahoo.co.jp/webapi/search/

text contents, where text contents are nodes and textual similarity between text nodes is the weight of the edge. Then the centrality of the text nodes is calculated by using the graph. The system measures $p(d)$ as the typicality of each disputed sentence $d \in D = \{d_1, d_2, \ldots, d_n\}$ using the LexRank algorithm. This computation is formalized as follows:

$$\mathbf{T} = \alpha \mathbf{S}^* \times \mathbf{T} + (1 - \alpha)\mathbf{p}, \quad \text{where } \mathbf{p} = \left[\tfrac{1}{n}\right]_{n \times 1} \tag{3}$$

Here, $\mathbf{S}^*$ is the column normalized adjacency matrix of the similarity matrix $\mathbf{S}$, where each $\text{sim}(d_i, d_j)$ denotes the cosine similarity between disputed sentences d_i and d_j. $\mathbf{T}$ is the typicality score vector, where each $t(d)$ denotes the typicality of disputed sentence d on the Web. We approximate $p(d)$ by $t(d)$.

   In practice, $p(d|q)$ of disputed sentence $d \in D$ is calculated as follows:

1. Collect disputed sentences $D = \{d_1, d_2, \ldots, d_n\}$ from the Web (as explained in Section 3.1).
2. Calculate $p(q|M_c)$ and $p(q|M_d)$ of all $d \in D$ using eq. 2. In this process, the system ignores stopwords in the collected disputed sentences.
3. Create feature vectors of the disputed sentences by ignoring stopwords. To weight terms of the vectors, the system uses $tf\text{-}idf$ weighting.
4. Calculate typicality $t(d)$ as $p(d)$ of all disputed sentence $d \in D$ using 3. The system recursively calculates the typicality scores T of all disputed sentences D until the scores are converged.
5. Calculate $p(d|q)$ of all $d \in D$ using the results of step 2 and 4 on eq. (1).

4   Experiment

We conducted an experiment to examine how effective our method is for enhancing users' credibility judgment in the Web search process. In this experiment, we used the following three viewpoints to evaluate our disputed sentence suggestion and compared our method with two baseline methods:

- **Relevance:** How relevant are suggested disputed sentences to a given query?
- **Alertingness:** How critical will users become about Web search results for a given query when they look at suggested disputed sentences?
- **Usefulness:** How useful are suggested disputed sentences in searching for additional evidence to judge the credibility of Web search results for a given query?

We compared our method with two baseline methods. One is a conventional keyword query suggestion method, which suggests related queries based on users' query logs (called **CKS**). To imitate the conventional keyword query suggestion, we issued each of the prepared queries into Yahoo Web Search[5] and used its suggested keywords. As for ranking order of suggested queries, we simply used the order in which Yahoo Web Search suggested the queries. The other baseline method is the typicality-based disputed sentence suggestion method, which focuses on only $p(d)$ in the process we explained

---

[5] Yahoo Web Search: http://search.yahoo.com/ and then manually obtained suggested queries.

Table 2. Query set

Category	Query
Controversy	dinosaurs extinction, global warming, earthquake cause
	plastic recycling, renewable energy
Health	effective diet, cancer treatment, mobile phone health
	vaccine side effects, caffeine overdose
Misunderstanding	light bulb inventor, telephone inventor
	civil war reason, caesar salad name
Food	coffee health, grapefruit seed extract benefit, potato poison
Finance	forex risk, mortgage refinancing, personal debt reduction

in Section 3 (called **TDS**). To execute disputed sentence suggestion, we set our method (called **Ours**) and **TDS** to collect 40 Web search results for each of the given queries. In addition, we set $\lambda = 0.5$ on eq. (1) for **Ours**.

4.1   Metrics for Evaluation

To evalate the effectiveness of **Ours**, **TDS**, and **CKS** from the viewpoints of relevance, usefulness, and alertingness, we checked how many relevant/alerting/useful suggestions appeared in top-k rankings. To evaluate them, we used the following equations:

$$r@k = \frac{1}{|Q|} \sum_{q \in Q} \frac{|Relevant(S_q(k))|}{k} \tag{4}$$

$$a@k = \frac{1}{|Q|} \sum_{q \in Q} \frac{|Alerting(S_q(k))|}{k} \tag{5}$$

$$u@k = \frac{1}{|Q|} \sum_{q \in Q} \frac{|Useful(S_q(k))|}{k} \tag{6}$$

Here Q means a query set and $S_q(k)$ means top-k suggested keywords/sentences that each algorithm outputs for the input query q. $|S|$ indicates the size of the set S. $Relevant(S)$, $Alerting(S)$, and $Useful(S)$ respectively mean the suggested keywords/sentences in S judged as relevant, alerting, and useful by least two human evaluators.

4.2   Participants and Materials

We recruited three human evaluators for this experiment. They all had experience in using Web search engines. For this experiment, we manually prepared 20 queries from five categories. The queries are shown in Table 2.

4.3   Procedure

In this experiment, we input each of the queries in Table 2 to the three methods (**Ours**, **TDS**, and **CKS**) and showed the suggested keywords/sentences to the evaluators. Then
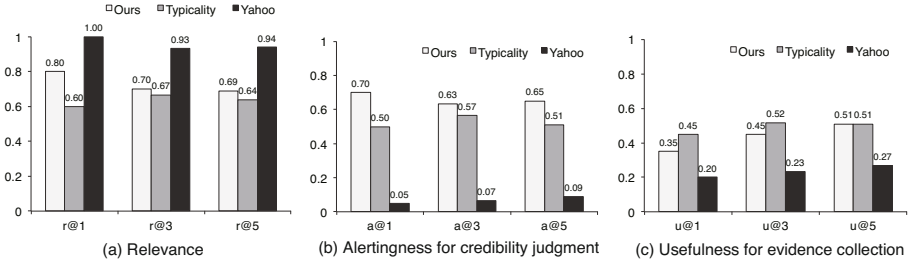
Fig. 3. Relevance, alertingness, and usefulness of three methods

we asked them to evaluate the suggested keywords/sentences in terms of relevance, alertingness, and usefulness. Before the evaluators evaluated suggested keywords/sentences for each query, we showed them a brief description like:

> *Suppose that you input the keywords "mobile phone health" to a Web search engine, and then the search engine shows some Web search results. In addition to the search results, the search engine also suggests some keywords/sentences as highlighted below. Now you are concerned about the credibility of the Web search results the search engine returns for the input keywords "mobile phone health". On the basis of the suggested keywords/sentences, please answer the following three questions.*

In each evaluation task for **CKS**, we showed the evaluators raw keyword queries suggested by Yahoo. In each evaluation task for **Ours** and **TDS**, we added the expression *"Some people doubt:"* to each of disputed sentences and showed the combined sentences to the evaluators (e.g., if the system found the disputed sentence *"mobile phone has a health risk"* for the query *"mobile phone health"*, the system suggested *"Some people doubt:x mobile phone has a health risk"* to the evaluators).

To evaluate the relevance, alertingness, and usefulness of each of the suggested keywords/sentences, we asked the evaluators to answer the following three questions with YES or No:

– **Relevance:** *Are suggested keywords/sentences related to a given query?*
– **Alertingness:** *Will you become more critical of Web search results about a given query when looking at suggested keywords/sentences?*
– **Usefulness:** *Are suggested keywords/sentences useful in searching for additional evidence to judge the credibility of Web search results about a given query?*

4.4   Results

Fig.3 shows the averages of relevance scores, usefulness scores, and alertingness scores for each method on a query set, using equations $r@k$, $a@k$, and $u@k$. According to Fig.3(a), in terms of relevance, **CKS** performed much better than **Ours** and **TDS**. However, **Ours** was not so bad at suggesting relevant sentences ($r@1 = 0.80$, $r@3 = 0.70$, and $r@5 = 0.69$). **CKS** is based on query logs a lot of users issued in actual Web search

sessions. Therefore, **CKS** can precisely suggest relevant keywords for given queries, unlike **Ours** and **TDS**, which analyze documents to suggest disputed sentences. The relevance performance of **Ours** was higher than **TDS** on $r@1$, $r@3$, and $r@5$. Especially, according to $r@1$, **Ours** was higher by 33% than **TDS**. We think that query likelihood $\prod_{t \in q}((1 - \lambda)p(t|M_c) + \lambda p(t|M_d))$ in eq. (1) contributes to the relevance score.

In terms of alertingness, Fig.3(b) shows that with **Ours** and **TDS**, at least 50% of suggested sentences in the top 5 rankings called the evaluators' attention to credibility judgment. The performance of **Ours** was higher than that of **TDS** on $a@1$, $a@3$, and $a@5$. On the other hand, as the result for **CKS** indicates, the evaluators did not become more careful of their credibility judgment at all, even when looking at the keywords suggested ($a@1$=0.05, $a@3$=0.07, $a@5$=0.09). This result is natural because the objective of conventional keyword suggestion is different from that of our disputed sentence suggestion, which aims at alerting users to suspicious information.

Fig.3(c) shows that the two methods that suggest disputed sentences performed better than **CKS**. This indicates that the disputed sentence suggestion can be more useful in searching for additional clues for credibility judgment on Web pages about given queries than conventional keyword suggestion.

Before this experiment, we expected that disputed sentence suggestion would be more useful in searching for clues for credibility judgment. However, neither of the two disputed sentence suggestion methods showed show usefulness scores as high as we expected. In addition, $u@1$ and $u@3$ of **Ours** were less than those of **TDS** (scores of the two methods were the same on $u@5$). We interviewed the evaluators after the experiment. Below is a typical evaluator comment:

> *The suggested disputed phrases themselves were often useful as evidence or clues to judge the credibility of Web pages about given queries. So I judged that I didn't need to search for Web pages using the suggested disputed phrases.*

This indicates that disputed sentence suggestion was not necessarily useless, although Fig.3(c) indicate that it was not so useful in searching for additional credibility evidence from the Web. We think that if systems can suggest disputed sentences relevant to given queries, the disputed sentences themselves are likely to be useful clues for credibility judgment on Web pages for given queries. The experimental results shows that **Ours** suggested more relevant disputed sentences than **TDS**.

Finally, these experimental results indicate that, by suggesting disputed sentences with reasonable relevance to given queries, our proposed method can make users more critical in the Web search process than the baseline methods can.

4.5   Case Study on Disputed Sentence Suggestion

Table 3 and 4 show some examples of good and bad disputed sentences the system suggested for given queries using our proposed method. We manually checked the suggested disputed sentences the evaluators judged as bad and categorized them into four classes: *non-relevant*, *ambiguous*, *difficult to understand*, and *grammatical error*.

As evidenced by sentences like *"potatoes are fattening"* for query *potato poison*, the system sometimes suggested non-relevant disputed sentences to given queries. We

Table 3. Good examples of suggested disputed sentences. Numbers in parentheses are ranking orders of suggested disputed sentences.

Query	Suggested disputed sentence
potato poison	*"green potatoes are poisonous"* (1)
vaccine side effects	*"tamiflu does not have potential side effects"* (5)
caesar salad name	*"the salad takes its name from julius caesar"* (1)
civil war reason	*"slavery was the cause of the civil war"* (1)
earthquake cause	*"fracking causes earthquakes"* (3)

Table 4. Bad examples of suggested disputed sentences and error types. Numbers in parentheses are ranking orders of suggested disputed sentences.

Query	Suggested disputed sentence	Error type
potato poison	*"potatoes are fattening"* (5)	Non-relevant
global warming	*"global warming was the cause"* (1)	ambiguous
telephone inventor	*"agb was not the inventor"* (3)	Difficult to understand
plastic recycling	*"shipping plastics for recycling"* (2)	Grammatical error

think this is because even if extracted disputed sentences are not very relevant to given queries, our system gives them high scores if they are typical on the Web. We have to think about a way to balance query likelihood scores and typicality scores of disputed sentences on eq. (1).

Even when the system maintained the relevance of disputed sentence suggestion, ambiguous disputed sentences were sometimes suggested, such as *"global warming was the cause"* for query *global warming*. In our method, the system simply extracts disputed sentences that appear behind linguistic patterns like *"it is not true that"* and *"no proof that"*. Therefore, the system often suggests that ambiguous and difficult-to-understand disputed sentences without context be extracted from Web search results.

The evaluators misunderstood that the system suggested some meaningless disputed sentences for given queries, although the disputed sentences were actually relevant to the queries and alerting. One possible reason is that the evaluators could not understand the meaning of some terms in the disputed sentences because they were not familiar with the sentences. For instance, the system suggested disputed sentence *"agb was not the inventor"* for query *telephone inventor*. If users do not know that term *agb* means *Alexander Graham Bell*, they may think the disputed sentence is useless and then will not become more critical of Web pages for the given query. It is important to consider the comprehensibility of sentences when suggesting them to users.

The system also sometimes suggested disputed sentences that were grammatically difficult to read like *"someone doubts: shipping plastics for recycling"* for query *plastic recycling*. Our system splits sentences with punctuation marks and extracts disputed sentences by simply focusing on declarative content clauses (*that clauses*). Therefore, the system often fails to extract grammatically complete sentences. We need deeper natural language processing to handle this problem.

5   Conclusion

In this paper, we addressed a new type of query suggestion to call users' attention to credibility judgment of Web search results for given queries in the Web search process. Given a query, such as *mobile phone health*, our system searches for statements some Web pages dispute about the query and suggests ones like *"mobile phone are bad for your health"* to users. While conventional keyword query suggestion focuses on making it easy for users to search for Web pages that match their search intent, our disputed sentence suggestion focuses on making them aware of the existence of suspicious information or to get clues for a credibility judgment on Web pages about their input query. Even if users are not concerned with Web information credibility and do not know about the existence of suspicious Web information, they can notice suspicious information and become more critical in searching for credible Web pages.

The experiment results show our method suggested effective disputed sentences to help users more critically search for credible Web pages, although the method had lower performance in terms of relevancy than conventional keyword suggestion. To improve our disputed sentence suggestion, we first have to improve the relevance of suggested disputed sentences by optimizing the balance between the query likelihood scores and typicality scores of them. In addition, we need to think about a way to suggest comprehensible disputed sentences because sometimes users cannot understand the meaning of suggested disputed sentences even if they are relevant and typical to given queries. Another important issue is how to support users' search for credible Web pages efficiently using suggested disputed sentences.

To safely and efficiently obtain information from the vast Web, search systems focusing on credibility will become more important in the future . We believe our proposed system can contribute to credibility-oriented Web search.

References

1. Sillence, E., Briggs, P., Fishwick, L., Harris, P.: Trust and Mistrust of Online Health Sites. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2004), pp. 663–670 (2004)
2. Nakamura, S., Konishi, S., Jatowt, A., Ohshima, H., Kondo, H., Tezuka, T., Oyama, S., Tanaka, K.: Trustworthiness Analysis of Web Search Results. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 38–49. Springer, Heidelberg (2007)
3. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998), pp. 275–281 (1998)
4. Liu, Y., Gao, B., Liu, T.Y., Zhang, Y., Ma, Z., He, S., Li, H.: BrowseRank: Letting Web Users Vote for Page Importance. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008), pp. 451–458 (2008)

5. Cui, H., Wen, J.R., Nie, J.Y., Ma, W.Y.: Probabilistic Query Expansion Using Query Logs. In: Proceedings of the 11th International Conference on World Wide Web (WWW 2002), pp. 325–332 (2002)
6. Boldi, P., Bonchi, F., Castillo, C., Vigna, S.: From "Dango" to "Japanese Cakes": Query Reformulation Models and Patterns. In: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI 2009), pp. 183–190 (2009)
7. Kotov, A., Zhai, C.: Towards natural question guided search. In: Proceedings of the 19th International Conference on World Wide Web (WWW 2010), pp. 541–550 (2010)
8. Suryanto, M.A., Lim, E.P., Sun, A., Chiang, R.H.L.: Quality-Aware Collaborative Question Answering: Methods and Evaluation. In: Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM 2009), pp. 142–151 (2009)
9. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: Proceedings of the 20th International Conference on World Wide Web (WWW 2011), pp. 675–684 (2011)
10. Schwarz, J., Morris, M.: Augmenting web pages and search results to support credibility assessment. In: Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems (CHI 2011), pp. 1245–1254 (2011)
11. Yamamoto, Y., Tanaka, K.: Enhancing Credibility Judgment of Web Search Results. In: Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems (CHI 2011), pp. 1235–1244 (2011)
12. Pirolli, P., Wollny, E., Suh, B.: So You Know You're Getting the Best Possible Information: A Tool that Increases Wikipedia Credibility. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI 2009), pp. 1505–1508 (2009)
13. Ennals, R., Trushkowsky, B., Agosta, J.M.: Highlighting Disputed Claims on the Web. In: Proceedings of the 19th International Conference on World Wide Web (WWW 2010), pp. 341–350 (2010)
14. Kawahara, D., Inui, K., Kurohashi, S.: Identifying contradictory and contrastive relations between statements to outline web information on a given topic. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), pp. 534–542 (2010)
15. Yamamoto, Y., Tezuka, T., Jatowt, A., Tanaka, K.: Supporting Judgment of Fact Trustworthiness Considering Temporal and Sentimental Aspects. In: Bailey, J., Maier, D., Schewe, K.-D., Thalheim, B., Wang, X.S. (eds.) WISE 2008. LNCS, vol. 5175, pp. 206–220. Springer, Heidelberg (2008)
16. Ennals, R., Byler, D., Agosta, J.M., Rosario, B.: What is disputed on the web? In: Proceedings of the 4th Workshop on Information Credibility (WICOW 2010), pp. 67–74 (2010)
17. Erkan, G., Radev, D.: LexRank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research 22, 457–479 (2004)

Exploration and Visualization of Administrator Network in Wikipedia

Jamal Yousaf[1], Juanzi Li[1], Haisu Zhang[2], and Lei Hou[1]

[1] Department of Computer Science and Technology Tsinghua University Beijing, China
[2] PLA University of Science and Technology Nanjing, China
{jamalyousaf,ljz,houlei}@keg.cs.tsinghua.edu.cn
zhanghaisu@139.com

Abstract. Wikipedia has become one of the most widely used knowledge systems on the Web. It contains the resources and information with different qualities contributed by different set of authors. A special group of authors named administrators plays an important role for content quality in Wikipedia. Understanding the behaviors of administrators in Wikipedia can facilitate the management of Wikipedia system, and empower some applications such as article recommendation and expertise administrator finding for given articles. This paper addresses the work of the exploration and visualization of the administrator network in Wikipedia. Administrator network is firstly constructed by using co-editing relationship and six characteristics for administrators are proposed to describe the behaviors of administrators in Wikipedia from different perspectives. Quantified calculation of these characteristics is then put forwarded by using social network analysis techniques. Topic model is used to relate content of Wikipedia to the interest diversity of administrators. Based on the media wiki history records from the January 2010 to January 2011, we develop an administrator exploration prototype system which can rank the selected characteristics for administrators and can be used as a decision support system. Furthermore, some meaningful observations are found to show that the administrator network is a healthy small world community and a strong centralization of the network around some hubs/stars is obtained to mean a considerable nucleus of very active administrators that seems to be omnipresent. These top ranked administrators ranking is found to be consistent with the number of barn stars awarded to them.

Keywords: Wikipedia, social network analysis, human factors, visualization.

1 Introduction

Wikipedia provides a fast and flexible way for anyone to create and edit encyclopedia articles. Since its creation in 2001, Wikipedia has grown rapidly into one of the largest online knowledge bases containing 3,632,075 articles. Wikipedia is attracting more than a million of visits daily and its huge success in turn leads to an immense research interest for better understanding of collaborative knowledge. Since every single revision is recorded and the collaborative knowledge building process is well

documented, a special group of users named Wikipedia administrators are still required to be very active and efficient. The administrators constantly observe and monitor the editing process of certain range of articles on voluntarily basis and normally can edit, delete or restore the original contents, remove or block the malicious users. According to the statistics of May 8, 2011, among 14,507,035 registered users, there are about 1,785 administrators. Therefore the role of the admin users becomes very important and more responsible because their efforts guarantee the trust and preservation of the original contributions. Therefore, it is necessary for us to get insight of the administrator network to observe and study the ongoing processes and to see if there exist some interesting patterns and phenomenon of them. Knowing the characteristics of admin users can bring many interesting applications for example the Wikipedia could automatically know the state of administrator's work and recognize the distribution of interesting topics of them. Also, this information can be used to recommend the articles to the proper admin users.

Though the Wikipedia's greatest success has gained much research on various aspects of Wikipedia, but little research focused on administrator's behavior in Wikipedia. Some of the research aims at validating some laws like power law, network properties in different kinds of networks, some concentrates on the trust building of the Wikipedia contents and the reputation system of the co-authors and others on social network analysis. There is a lack of the systematic research on characterizing the work of administrators in Wikipedia. To the best of our knowledge no work could reflect the overall features of the administrator's in Wikipedia. This paper addresses the following questions. 1) How could we define the behaviors of the admin users in Wikipedia? 2) How could we quantify the behaviors of the admin users so that we can better understand them in Wikipedia? 3) Could we provide an exploratory and visualized tool to give comprehensive description of the admin user in Wikipedia?

In this paper we propose a systematic definition of administrator's characteristics and contribution of this paper could be summarized as follows:

a. **Systematic definition for administrator characteristics**
   Administrators are the normal users with some additional responsibilities and their work guarantees the trust on the contents and hence increases the reliability of the text. In this paper we define six features to reflect the characteristics of administrator's work from different respects including diversity of the admin user interests, the influence & importance across the network, and longevity & activity in terms of contribution.

b. **Quantitative calculation by using social network analysis technology**
   We give the quantitative calculation methods for each characteristic, and the overall feature measuring method. The main idea is to provide the precise and robust measure for measuring the admin user efficiency and activeness. The important thing is that these measures capture not only the quantity of contributions but also relate the contents of Wikipedia to the admin users.

c. **Visualization of administrator characteristics**
   All these features are collectively implemented in an administrator exploration prototype system, which combines all the features and computes the overall rank. It shows the overall administrator interests and contribution, strengths and ranks the intensity of contributions in terms of efficiency and activeness.

The rest of the paper is organized as follows. In section 2, we introduce the related work. In section 3, we present the initial observations about the network structure and statistical observations. In section 4, we define the characteristics of administrators from different views, and give the corresponding measuring methods. In section 5, we present the implementation of the prototype system, some meaningful observations and give the analysis of administrator's network evolution. Finally, Section 6 concludes this paper with future research.

2   Related Work

Co-authorship network, an important form of social network, has been intensively studied in this movement ([26] 2001b; [24] 2002; [22] 2003; [21] 2004; [13] 2005; [11] 2006; [9] 2007; [6] 2008; [4] 2009; [3] 2010; [2] 2011; [1] 2011). Many efforts have been made by the organizations to identify metrics useful for assessing the level of scientific productivity of single author, research groups or institutes on a local, regional or international scale [17]. There are generally three schools of underlying techniques that support visualizing co-authorship, such as co-authorship network [12], [19], [25], social network analysis [15], and small world network [21]. These techniques are based on the principles of statistics, graph theory, psychology and even a combination. In this section we will review the related work on analysis of co-authorship in Wikipedia and other academic domains and the scale free networks.

When writing on public wiki sites such Wikipedia, however, co-authorship emerges as an implicit relationship from working on the same article, rather than being planned from the outset as it is the case in traditional collaborative writing. In the paper (Yan, E. 2010) [3], it discussed the usability of centrality measures in author ranking, and suggest that centrality measures can be useful indicators for impact analysis. Co-authorship of a paper can be thought as the documenting collaboration between two or more authors [19]. General properties of the Wikipedia system and its user community have been analyzed by (Ortega, F., 2007) [9]. Other research has been done on analyzing article quality in Wikipedia and found that the cooperative content production model of Wikipedia results in high article quality [7]. In the paper (Libyy Veng-Sam Tang, 2008) [6], it presented new method for calculating the degree of co-authorship for a given pair of authors. This method is more accurate than any other existing methods that analyze co-authorship and has the satisfactory performance that makes suitable for online use.

In academic context the co-authorship relationships are perhaps one of the most important types of connections between scholars.   In the paper (Serge, Galam. 2011) [2], proposed the Tailor Based Allocation (TBA) for multiple authorships, which may varies from one discipline to another. Researchers have shown great interests in analyzing co-authorship networks specific to their research communities to shed light on the characteristics of their scientific collaboration, including physics [26], biomedicine [26], mathematics [24], management [11], databases [22], and digital libraries [13]. Some statistics are proposed to describe the features of authors such as academic research social network mining system –arnetminer [28]. Several generic properties concerning co-authorship networks in various fields have been identified.

Most of the real-world networks having power law degree distribution and are, thus called scale-free networks. One of the first extensive quantitative studies on the Wikipedia community was presented in (J. Voss, 2005) [14], where its growth is shown to follow an exponential trend. The main characteristic of the scale-free network is the presence of hubs that concentrate collaborations. This approach is very useful for identifying the role of authors that we could define as "key players" of the network according to their degree, closeness and betweenness centrality [3]. It is also remarkable that one can find very different collaboration patterns depending on the scientific disciplines. Tightly linked communities give rise to all the benefits of social cohesion, such as the timely diffusion of new information, the creation of a common culture and the enforcement of social norms [4].

3 Preliminaries

3.1 Administrator Network

An administrator network is a network where the vertices are admin users and the edges represent the editing of the common articles between two admin users. If two admin users are the authors of the same article, that is, they have at least one revision each belonging to the same article. This co-editing relationship is a relationship between the authors working on the same article. Figure 1 shows that administrator network of five admin users; each one of them has contributed to some articles.
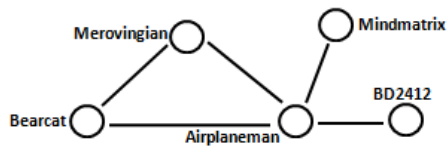
Fig. 1. Co-authorship of Administrators Network

3.2 Initial Observations

Before proceeding, we first engage in some high-level investigation of about the network structure and other statistical observations. This will help us to further explore and monitors the true observations by analyzing the different scenarios on the Wikipedia network. In the following we give some initial observations about the administrator's network.

Diameter and Clustering Coefficient
The diameter and the clustering coefficient of the administrator's network were computed to examine the structure of the network at macro level. Since more robust measure of the pair-wise distances between nodes of a graph is the effective diameter [26]. It is observed that the diameter of the admin-users' network is about the 4 hops and the effective diameter is about the 2 hops as shown in the Figure 2. This shows that how two random pair of admin users can be reached within certain numbers of

hops among the administrator's who communicate or performs editing together. These short distances can be explained in virtue of the lower barriers to the collaboration between the admin users and it can also be interpreted as an effect of centralization of the network around some active users.

The Clustering coefficient is the measure of transitivity in the network especially on the social network [27]. The transitivity in the network is observed by plotting the average clustering of the whole network as shown in the Figure 2 with average clustering coefficient 0.6051 respectively. The value of the average clustering decreases with the increase in degree of the node in the admin user graph. It is similar like in other real networks the average clustering decreases as the node degree increases because of the fact that low-degree nodes belong to very dense sub graph and those sub graphs are connected through the hubs. In our analysis the decreasing trend of the clustering coefficient can also be seen as a symptom of the growing centralization of the network. This shows that a strong centralization of the network around some hubs/stars, a considerable nucleus of very active users who seems to be omnipresent.
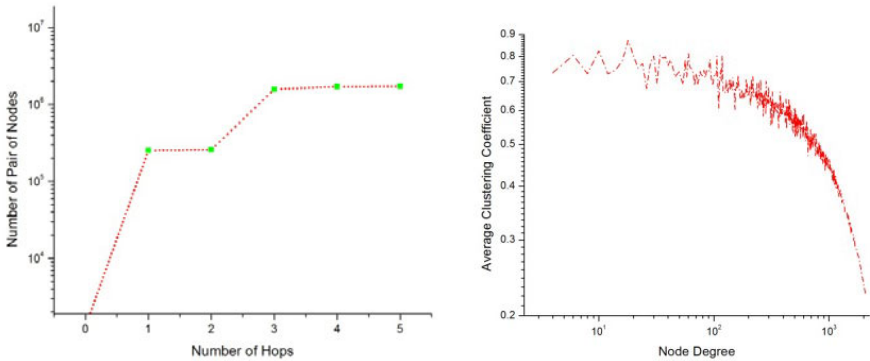
Fig. 2. Distribution of Path length in hops & Trend of Clustering Coefficient

Lotka's Law

The Lotkas' Law describes the frequency of the publication by "the number of authors making n contribution is about $1/n^2$ of those making one and proportional of all contributors that make a single contribution, is about 60 percent" (Lotka, A. J. 1926). The Lotkas' Law is also observed on Wikipedia in relation between admin users and the number of terms they edited during the specific period of time. It has been observed that very few admin users have edited large number terms as shown in the Figure 3. This distribution follows the trend approximated by the equation as: $y = \frac{C}{x^n}$ where $n \approx 1.7$, and C is the constant. Moreover about 70 percent of authors have edited less than 5 percent of the maximum number of terms edited by admin users.

Summary: In summary, the administrator network is a healthy small world community having a small average distances and a strong centralization of the network around some hubs/stars is observed. This shows a considerable nucleus of very active administrators who seems to be omnipresent. The Lotkas' law observation

shows two important processes, one is that there are group of the admin users who are very active and having large number of edited terms and secondly this also shows that the work needed to maintain the high quality is rapidly increasing.
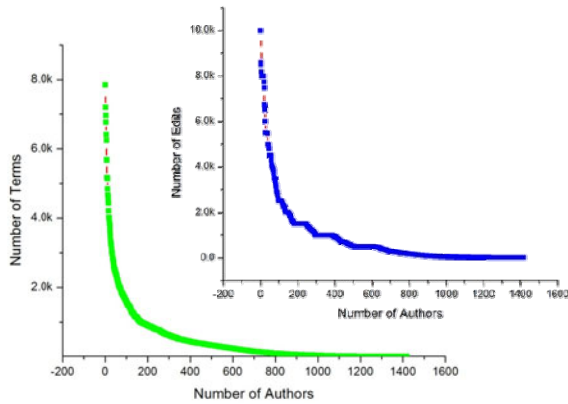
Fig. 3. Administrators with edited terms

These initial observations validates that there are hubs/stars nodes with high degree present inside the network which are very active and efficient and also the short distances among them depicts the lower barriers to the collaboration between any pairs of admin users.

4   Administrators' Characteristics

The administrator's characteristics can be best commented by first looking at the tasks performed by them. The approach and the nature of the work of the administrators' are quite different than the normal users of the Wikipedia. In this section, we define some features to characterize these kinds of abilities for the administrators.

4.1   Influence

Normally, users decide to adopt activities based on the activities of the people they are currently interacting with and then simultaneously form new interactions as a result of their existing activities. All these directly relate to the social influence. Similarly in large social graph, nodes are influenced by the other nodes for various reasons. Influence push the system to the uniformity of the behavior and users are similar to their neighbors because of the social influence. Generally vertices with higher degree or more connections are more central to the structure and tend to have a greater ability to influence others. The Degree centrality equals the number of the ties that a vertex has with the other vertices. For a graph $G = (V, E)$ with 'n' vertices, the degree centrality $C_D(v)$ for a vertex is:

$$C_D(v) = \frac{\deg(v)}{n-1} \tag{1}$$

4.2   Importance

An interesting question is whether these very influential admin users are preferentially linked with other highly important ones or not. In many circumstances the vertex's importance in a network is increased by having connections to the other vertices that are themselves important. Eigenvector is based on the principle that the importance of a vertex depends on the importance of its neighbors. It is proportional to the sum of the centralities of the vertex neighbors, so that vertex can have high centrality either by being connected to a lot of others or being connected to others that are highly central themselves. Here we use it as the measure to describe the importance characteristic of administrator. The formula in equation (2) shows that the prestige v_i of a vertex is proportional to sum of the prestige of the neighboring vertices pointing to it:

$$\lambda x_i = \sum_{j:j \to 1} x_j = \sum_j A_{ij} x_j = (A'x)_i \qquad (2)$$

Where x_i is the ith component of the eigenvector of the transpose of the adjacency matrix with Eigen value λ. Eigen Centrality depends upon both the number and the quality of connections.

4.3   Longevity

The admin users reported that it was difficult for them to catch up with overall activity of their work after a long absence (e.g. a few weeks long vacation). The longevity reflects the length of the each admin user contribution throughout the specified period. We consider the number of days d_k as the basic unit for this measure, the first day considered to be the first edit day of each user. The number of days reflects the seriousness and the level of commitment of admin user to his job. The most important part of this definition is the comparative nature of it and as the longevity is the measure of the striving effort of the users' quality work and can be mathematically defined as:

$$\text{Longevity } (v_i) = \sum_{k=1}^{n} d_k \qquad (3)$$

Where n is the total number of days worked by each admin user during the specific period.

4.4   Activity

Admin user time is mostly spent on the patrolling: detecting usual activities and tracking the vandalism. They control the content of the new articles, of long and numerous revisions on single article or from single user, as well as revision from the new user or unknown users. The activity of admin users is simply defined as the measurement of work done or delivered at the lowest level in each month over specific period of the time. It reflects the state and quality of the being active among the other admin users throughout the period.   It can be defined mathematically as:

$$\text{Activity } (v_i) = \sum_{j=1}^{m} \left(\frac{\sum_{k=1}^{n} E_k}{\left| \sum_{k=1}^{n} E_k \right|} \right)_j \qquad (4)$$

Where E_k is the edit done by the vertex i, m is the total number of the months and 'n' is the number of edits and $|\sum_{k=1}^{n} E_k|$ is maximum number of edits in that particular month by any vertex respectively.

4.5   Diversity

Diversity is a measure of performance across different research interests or the topics. Generally an expert may have several different research fields. Diversity reflects the quantitative measure of the degree of expertise across different topics of interests. It is very important to know the diversity of the admin users so that one can easily find the expert on particular topic. For an expert vertex v, the topic distribution $P_v(t)$ is defined as:

$$P_{v_i}(t) = \frac{\text{Number ofedit terms of } v_i \text{ belongs to topic t}}{\text{Total Number of Edit Terms of } v_i}$$

Where $t \in$ topics (v_i) and these set of topics are determined by Latent Dirichlet Allocation (i.e. probabilistic generative model) as explained in section 5.1.
The vertex diversity can be as the entropy of the above distribution $P_v(t)$:

$$\text{Diversity } (v_i) = -\sum_{t \in \text{ topics } (v_i)} \left(P_{v_i}(t) \log P_{v_i}(t) \right) \tag{5}$$

4.6   Interests

Interests of the admin user can be determined by navigating the edit history of the respective user. The terms and the articles the admin user have initiated or contributed will describe his interests and expertise. Let *m* documents are contributed by the admin user which are denoted by set $D = \{D_0, \ldots, D_{m-1}\}$. Each document is a set of n topics denoted by $T = \{T_0, \ldots, T_{n-1}\}$. Therefore the Interests of the admin user $I(v_i)$ are defined as:

$$I(v_i) = \max_k \left(P_{v_i}(t) \right) \tag{6}$$

Where k = 1 to 10 and gives the 10 top most topics of the interests of an admin user.

4.7   Overall Rank

Based on above definitions, we propose and develop the ranking system for the admin users of the Wikipedia. Using these results will be helpful in contributing the unified definition to evaluate the behavior of the admin users. The overall rank is calculated as:

$$\text{Rank } (v_i) = \frac{\sum_{j=1}^{5} \left(W_j \times M_j(v_i) \right)}{\sum_{j=1}^{5} W_j} \tag{7}$$

Where M_j & W_j are the measures and the weights of respective measures from equation (1-6). The weights are considered to be as unity for best results as shown in Figure 8.

5   Implementation

Data dumps of Wikipedia databases are available for download. This enabled us to evaluate our method using actual data. We have downloaded the 504,864 terms/articles and 890,299 numbers of edits are extracted from Wikipedia English language edition. The data dumps used by us are between the January, 2010 to January, 2011 period. We use the dataset to extract topics and perform the administrator evaluation of six characteristics.

5.1   Topic Extraction in Wikipedia Using Latent Dirichlet Allocation (LDA)

The Latent Dirichlet Allocation (LDA) has been introduced as a generative probabilistic model for a set of articles [23]. In LDA, each topic is represented by a probability distribution over the terms and each article is represented by a probability distribution over the topics. We used LDA to extract topics from the downloaded Wikipedia articles which were edited by the admin users. Figure 4 illustrates the process of calculating the interests and diversity of the admin user. This leads us in uncovering the group of expertise user on the some particular specialized area or topic. We extracted 200 topics in our experimentations.
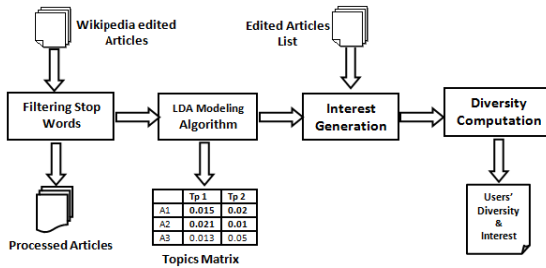
Fig. 4. Process of finding the interests and diversity of the admin user

5.2   Administrator Exploration Prototype System

A prototype application named Administrator Exploration Prototype System is developed that implements various measures as discussed in above section 4. All the code is written in modular form that adds new features and enhances the main functionality of the Media Wiki. With our extension the Media Wiki user can easily get the statistics, interests, expertise in his field and the rank of the admin users by just entering the admin user name. The user can view the rank list of the admin users based on the different measures like longevity, diversity or overall rank e.t.c. The Graphical User Interface is as shown in the Figure 5.

Graphical User Interface
The interface consists of four main parts depending upon the type and display of information. One part gets the admin user name and displays the six colorful bars whose lengths are proportional to score of the respective measures. These bars show

the percentage score of each measure based on his edit history record. Second part shows the month wise contribution of the admin user relative to the maximum number of edits in that particular month. The two colors bar chart (dark green and the dark blue) are used to show the user actual edit counts in that particular month and also shows his edit count with respect to the maximum edit counts in that month respectively. Third section shows the users list of interests and expertise topics. The fourth part shows the rank list of the admin users based on any one of the selected measure. All these four sections collectively quantify the full role of the admin user.

Expert Selection

The other important part of administrator exploration prototype system is expert finding on a particular topic as shown in Figure 6. Expert selection aims to determine
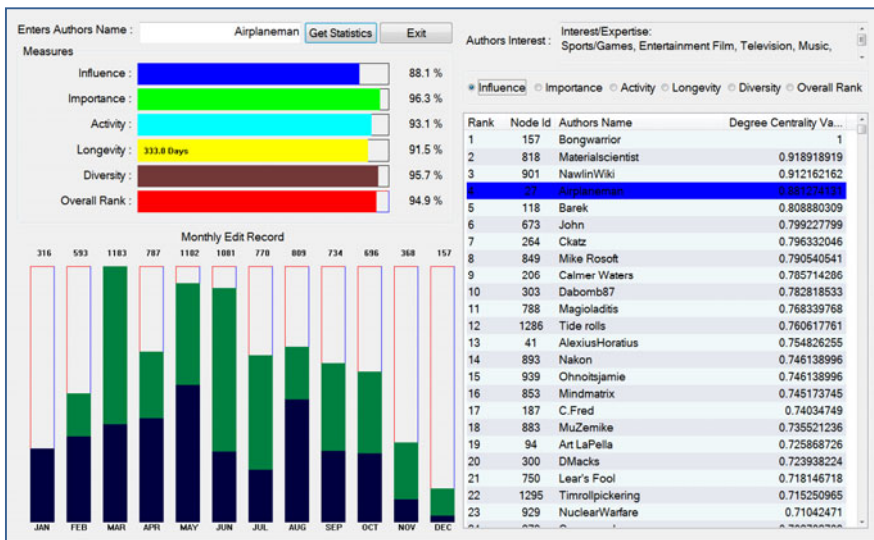
Fig. 5. Graphical User Interface of the Administrator Exploration Prototype System

Fig. 6. Expert User Search

the members of the expertise group. The user needs to submit the keyword or topic name for finding the expert admin user on Wikipedia administrator network.

Administrator exploration prototype system will automatically find the expert authors based on the editing history of each admin user. The resulting list displays the username and bar whose length is proportional to his degree of expertise on that particular topic. The top admin user bar is given the full length display and rest is normalized with score of the top author. The search window will shows top ten results.

5.3   Diversity

The diversity of the admin users as shown in Figure 7 is very well distributed and most of the admin users have very diverse interest. It also reflects the diverse editing capability of admin users across several different topics. In our analysis it seems from the histogram that the admin users have their watch list, which are actively maintaining by them with periodic addition. Most of these watch list are overlapped by the other admin users and because of that almost 85 percent of the terms are edited and rechecked by more than one admin user. This means that the editing has been closely observed and well monitored by admin users to ensure the quality and prevents the vandalism.
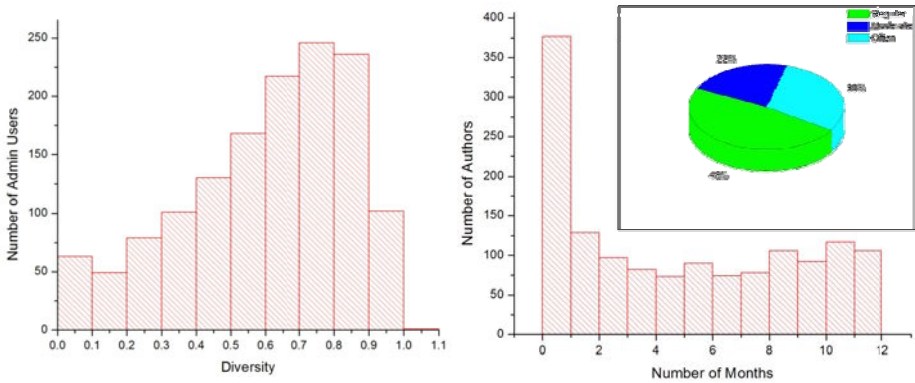
Fig. 7. Spread of diversity and the categories of the admin users

Looking at the trends and patterns of the admin users, it has been observed that there are almost 48 percent of admin users who are regular contributors of the Wikipedia and 22 percent are the admin users who work for 6 to 9 months. The last one is the 30 percent of the community who have contributed less than 6 months. We can divide them into three categories as regular, moderate and the often contributors to Wikipedia as shown in the Figure 7. In our analysis the regular users, a core of very active contributors play a fundamental role and continue playing by leveraging their centrality in growing network. The often users contribute less than 6 months but most of this category users have large number of edits with less longevity and their diversity is above average.

It is also observed that there are some inactive users in the network who didn't perform any activity for the whole year. It has also pointed earlier that it is very difficult for the admin user to catch up with overall activity of their work after a long absence even for few weeks long vacation. Therefore an automatic reminder should be issued to the individual admin user of such group after some period of absence.
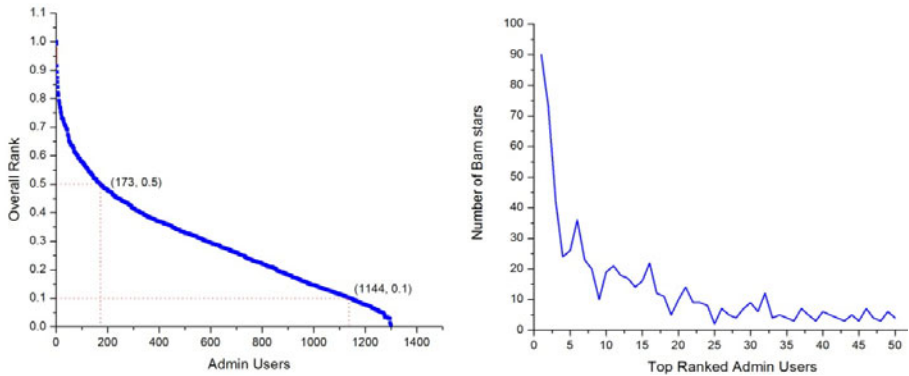
Fig. 8. Overall rank of the admin users and their number of barn stars

5.4   Overall Rank and Barn Stars

The overall rank of the admin users is shown in Figure 8. The ranking curve initially decays exponentially up to the rank 0.5 and then decays linearly up to the rank 0.1 with the increase in number of admin users and finally a very slow decrease at the end was observed. The exponential part shows that few percent of the admin users are most active and influential with high diversity among the rest of them. In our analysis these are the main core of active contributors of the Wikipedia and this group has the communal trust and confidence, rather than checklists and edits counts. The linear part consists of the large number of the admin users, whose average activity is less than the group of most active users. The last part of the curve with slow decreasing rate shows the small group of the admin users who often maintains their watch list on Wikipedia. By analyzing the profiles of the top ranked fifty admin users as a test case, it has been observed that the number of barn stars received by them also follows the similar trend as we overall ranked the admin users. This reflects the great confidence and validation of our approach and results.

6   Conclusions

Wikipedia has gained increasing importance and use throughout the world. Through its use, implicit groups of expertise are established around different set of topics. However, as these relationships are difficult to discern, data analysis techniques are used for discovering them. At first we have looked at the evolution of network properties like clustering co-efficient and the diameter. The initial results imply the evidence of strong

centralization of the network around some stars, a considerable nucleus of very active users who seem to be omnipresent. The high centrality of socio-metric stars points out the key role that "elite" continues to play in the community of Wikipedia, despite of the rapid growth of the number of common users. We have determined the interests of admin users and quantified the diversity in their interests, which would be very useful in determining and recommending some experts on given topics on Wikipedia. We have defined certain measures and applied it to analyze the social behavior and the dynamics of the English Wikipedia administrator community. We have defined certain measures and applied it to analyze the social behavior and the dynamics of English Wikipedia administrator community. We rank the admin users based on their contributions and interests on Wikipedia, which can be used as decision support system for administrators. By using this system an automatic reminder can be issued to the admin users after some period of absence. Future work will be to investigate the features which are used to assess trustworthiness of articles like textual features, references and images. This work will be baseline for future work.

References

1. Ding, Y.: Scientific collaboration and endorsement: Network analysis of coauthorship and citation networks. J. Informetrics, 187–203 (2011); Galam, S.: Tailor based allocations for multiple authorship: a fractional gh-index. Scientometrics 89(1), 365–379 (2011)
2. Yan, E., Ding, Y.: Applying centrality measures to impact analysis: A coauthorship network analysis. CoRR (2010)
3. Panzarasa, P., Opsahl, T., Carley, K.M.: Patterns and dynamics of users' behavior and interaction Network analysis of an online community. Journal of the American Society for Information Science and Technology 60(5), 911–932 (2009)
4. Tang, L.V.-S., Biuk-Aghai, R.P., Fong, S.: Method for measuring the co-authorship relationship in mediawiki. In: Int. Sym. Wikis. ACM (2008)
5. Rodriguez, M.A., Pepe, A.: On the relationship between the structural and socioacademic communities of coauthorship network. Journal of Informetrics, 195–201 (2008)
6. Wilkinson, M., Huberman, A.: Cooperation and Quality in Wikipedia. In: Proceedings of the 2007 International Symposium on Wikis, October 21-23, pp. 157–164 (2007)
7. Ortega, F., Barahona, M.: Quantitative Analysis of the Wikipedia Community of Users. In: Proceedings of the 2007 International Symposium on Wikis, pp. 75–86 (2007)
8. Vidgen, R., Henneberg, S., Naude, P.: What sort of community is the European Conference on Information Systems? European Journal of Information Systems 16(1), 5–19 (2007)
9. Acedo, F.J., Barroso, C., Casanueva, C., Galán, J.L.: Co-authorship in management and organizational studies: an empirical and network analysis. Journal of Management Studies 43(5), 957–983 (2006)
10. Yin, L., Kretschmer, H., Hanneman, R.A., Liu, Z.: Connection and stratification in research collaboration: An analysis of the COLLNET network. Information Processing and Management 42, 1599–1613 (2006)

11. Yoshikane, F., Nozawa, T., Tsuji, K.: Comparative Analysis of Co-authorship Networks Considering Authors' Roles in Collaboration: Differences between the Theoretical and Application Areas. In: ISSI 2005, vol. 2, pp. 509–516 (July 2005)
12. Liu, X., Bollen, J., Nelson, M.L.: Co-authorship networks in the digital library research community. Information Processing and Management 41, 1462–1480 (2005)
13. Voss, J.: Measuring Wikipedia. In: Proceedings International Conference of the International Society for Scientometrics and Informatics (2005)
14. Elmacioglu, E., Lee, D.: On six degrees of separation in dblp-db and more. ACM SIGMOD 34, 33–40 (2005)
15. Cheng, A., Friedman, E.: Sybilproof reputation mechanisms. In: Proc. of the ACM SIGCOMM Workshop on Economics of Peer-to-Peer Systems. ACM Press (2005)
16. Hirsch, J.E.: An index to quantify an individual's scientific research output. Proceedings of the National Academy of Sciences of the USA 102, 16569–16572 (2005)
17. Bryant, S.L., Forte, A., Bruckman, A.: Becoming Wikipedia: Transformation of participation in a collaborative online encyclopedia. In: Proceedings of SIGGROUP (2005)
18. Newman, M.E.J.: Coauthorship networks and patterns of scientific collaboration. P. Natl. Acad. Sci. USA 101(1), 5200–5205 (2004)
19. Hassan, A.E., Holt, R.C.: The Small World of Software Reverse Engineering. In: Proceedings of the 11th Working Conference on Reverse Engineering, November 08-12, pp. 278–283 (2004)
20. Kretschmer, H.: Author productivity and geodesic distance in bibliographic co-authorship networks and visibility on the Web. Scientometrics 60(3), 409–420 (2004)
21. Nascimento, M.A., Sander, J., Pound, J.: Analysis of SIGMOD's co authorship graph. ACM SIGMOD Record 32(3), 8–10 (2003)
22. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)
23. Barabasi, A.-L.: Linked-the new science of networks. Perseus Press, Cambridge (2002)
24. Chen, C., Paul, R.J.: Visualizing a Knowledge Domain's Intellectual Structure. IEEE Computer 34(3), 65–71 (2001)
25. Tauro, S.L., Palmer, C., Siganos, G., Faloutsos, M.: A simple conceptual model for the internet topology. In: GLOBECOM, vol. 3, pp. 1667–1671 (2001)
26. Newman, M.E.J.: The structure of scientific collaboration networks. Proceedings of the National Academy of Science of the United States of America 98(2), 404–409 (2001b)
27. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
28. Arnetminer, http://www.arnetminer.com

An Efficient Algorithm for Arbitrary Reverse Furthest Neighbor Queries

Jianquan Liu[1], Hanxiong Chen[1], Kazutaka Furuse[1], and Hiroyuki Kitagawa[1,2]

[1] Department of Computer Science, Graduate School of SIE, University of Tsukuba,
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8577, Japan
[2] Center for Computational Sciences, University of Tsukuba,
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8577, Japan
ljq@dblab.is.tsukuba.ac.jp,
{chx,furuse,kitagawa}@cs.tsukuba.ac.jp

Abstract. Given an object set O and a query object q, the reverse furthest neighbor (RFN) query retrieves the objects in O, whose furthest neighbor is q. In this paper, we consider the arbitrary RFN query that is without constraint of its location. The state-of-the-art method is not efficient for such kind of queries. Therefore, we address this problem by introducing our new findings on the filtering techniques. Firstly, we show the evidence that exhibits the inefficiency of the state-of-the-art method. We then figure out a non-trivial safe area to guarantee the efficiency for query processing, even meeting the ideal efficiency. We also design an efficient algorithm to answer the RFN query without any cost of filtering or refinement, when q is located in such safe area. Extensive experiments on both synthetic and real datasets are conducted to evaluate the effectiveness, efficiency and scalability of our algorithm. The results sufficiently indicate that our algorithm significantly outperforms the competitive ones in all the aspects.

Keywords: Similarity search, Reverse furthest neighbor, Convex hull, Safe area.

1 Introduction

Similarity search has been extensively studied in the recent decades. It often includes the familiar query types in terms of nearness, such as range query, k-nearest neighbor query (k-NN) [3, 6, 7, 11, 15, 18], reverse nearest neighbor query (RNN) [1, 10, 16, 17, 19], and so on. On the other hand, the opposite query types — "furthest" versions in terms of farness, such as k-furthest neighbor query (k-FN) and reverse furthest neighbor query (RFN) are lack of concentrative investiga-
tions. In this paper, we are interested in the RFN query.
Here, an RFN query is to retrieve the objects in an object
set O, whose furthest neighbor is the query q. For instance,
as shown in Fig. 1, given the object set O and the query ob-
ject q, then the answer to the RFN of q is $\{o_3, o_4\}$. Simply,
the processing computes all the pairs of distances between
q and o_i ($o_i \in O$). As illustrated by the dotted lines, object
o_3 holds the furthest distance to q than other objects in O
thus is included in the answer set. o_4 is an answer as the
same reason.

$O=\{o_1, o_2, \cdots, o_7\}, \quad q$
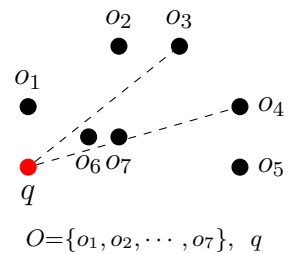
Fig. 1. Example of RFN

Regarding the applications of RFN, as pointed out by the scientists of operation research [2,5,8,9,13,14], the classical facility location problem that asks for the optimum location of a new facility (rubbish dumps, chemical plants, super market, police station, transmitter, etc.) is of critical importance in the real life. Especially, efficiently answering the problem of *competitive* facility location is still an attractive open problem [8]. Facing such research mission, efficient processing of RFN query must become significantly helpful. For simplicity, we highlight a real world example of competitive facility location problem as follows.

Application. Consider to open a new store for a convenience store franchise chain, such as 7-Eleven, Lawson, FamilyMart, etc.. It is one of the important factors to reduce the influence (competition) to the existing ones. Suppose that there are several location candidates ($Q = \{q_1, q_2, \cdots, q_m\}$) for opening a new store, and that the locations of existing ones ($O = \{o_1, o_2, \cdots, o_n\}$) are known. Under the consideration of competition, it is better to choose the location that is far from the existing ones in number as many as possible. It means that most of the existing stores take the chosen candidate as their furthest neighbor, thus the influence can be maximally reduced (i.e., $arg\max_{q_i \in Q}\{|\text{answers to } q_i's \text{RFN}|\}$).

In such kind of applications that take into account the evaluation criterion in terms of farness but not nearness, the RFN query plays an important role to help to make decisions. Nevertheless, due to the different distance properties between farness and nearness, the existing approaches concerning nearness cannot be employed to solve the corresponding problems directly. This implicates that specialized new algorithms should be exploited to handle the RFN query.

However, to the best of our knowledge, throughout the recent decades, there are few discussions concentrating on the query type of RFN. Yao *et al.* [20] are the first who fully identified the RFN query and proposed several algorithms to handle the query processing, such as the so-called "progressive furthest cell (PFC)" and "convex hull furthest cell (CHFC)" algorithms. The both algorithms adopt R-tree index, and use furthest Voronoi cell (fvc) to determine whether the objects $o \in O$ are q's RFN. In spite of that their algorithms are reported much efficient than brute-force search (BFS), they still need time consuming construction of dynamic convex hull and the furthest Voronoi cell w.r.t query q on the fly. Against these drawbacks, the state-of-the-art method is proposed in our previous work [12]. The so-called "PIV" algorithm processes the RFN query in a new workflow that consists of *filtering* and *refinement* phases, utilizing the efficient properties derived from triangular inequality based on the representative pivots. However, for an arbitrary query without any location constraint, the PIV algorithm is not always powerful due to the unnecessary cost for filtering.

Motivating us to further accelerate the RFN query processing on the fly, in this paper, we introduce our new findings on the filtering techniques, that adapt for arbitrary query and make the filtering cost dramatically decrease. Our findings and the main contributions of this work are summarized as follows.

1. We first analyze and clarify the cases when the PIV algorithm is inefficient.
2. Then, we figure out a non-trivial safe area to guarantee the efficiency. We also design an efficient algorithm (named PIV$^+$) to answer the RFN query without any cost of filtering and refinement, when query q is located in such safe area.

3. We discuss the complexity of the proposed algorithm, and perform extensive experiments to evaluate the effectiveness, efficiency and scalability.

The remainder of the paper is organized as follows. Section 2 surveys the related work, then preliminary knowledge used in this work is described in Section 3. Then we introduce the details of our new findings in Section 4, including the safe area, the mathematical proof, the algorithm and the complexity analysis. The experimental evaluations are presented in Section 5. Finally, Section 6 comes to conclude this work and goes into perspective of the future work.

2   Related Work

The query type of reverse furthest neighbor (RFN) was first slightly mentioned in [10], when the authors mainly discussed the reverse nearest neighbor (RNN) query in their work. However, throughout the past decades, there was few work focusing on the RFN query, until the recent work [20] appeared that is the work most closely related to ours.

In [20], Yao *et al.* originally defined the query type of RFN, and proposed the progressive furthest cell (PFC) algorithm and the convex hull furthest cell (CHFC) algorithm to handle RFN query. Both adopt R-tree index, and use the furthest Voronoi cell (fvc) to determine whether the points $o \in O$ are q's RFN. The $fvc(q, O)$ is to define a convex polygon w.r.t the query q in the given space of dataset O. To compute the $fvc(q, O)$, firstly draw the bisector line of each line segment oq ($o \in O$), then the space is separated into two subspaces, the $fvc(q, O)$ takes the intersection of all subspaces far away from the query q. The $fvc(q, O)$ strictly limits the answer set if and only if the point $o \in fvc(q, O)$. Straightforwardly, the authors proposed the PFC algorithm to compute $fvc(q, O)$ with the R-tree for each given query q on the fly. The authors also pointed out that the post-processing of PFC algorithm is expensive, so that they designed the faster one (CHFC) by utilizing the important property of convex hull. Instead of computing the $fvc(q, O)$ over the whole dataset, the CHFC algorithm constructs the dynamic convex hull C_O on the merged dataset $O \cup \{q\}$, computes the $fvc(q, O)$ only based on C_O, and then performs a containment query on R-tree index to retrieve the points contained in the $fvc(q, O)$ as answers to q's RFN. Hence, the CHFC algorithm outperforms the PFC, without expensive post-processing.

Although the CHFC algorithm was reported much efficient than brute-force search (BFS) and PFC, it still needs time consuming construction of convex hull (C_O) and the furthest Voronoi cell ($fvc(q, O)$) w.r.t query q on the fly. To overcome these drawbacks, the state-of-the-art approach was proposed in our previous work [12]. The approach independently pre-computes the static convex hull C_O of dataset O without considering the dynamic construction by merging q into dataset O, and selects the vertices of C_O as representative pivot set S_{piv} for the utilization of filtering. The pairwise distances between $\forall p \in S_{piv}$ and $\forall o \in O$ are pre-computed and stored into an metric index, among them, the maximum distance w.r.t each o is recorded as well. A so-called "PIV" algorithm was proposed to handle the RFN query in a new workflow that consists of *filtering* and *refinement* phases. The filtering phase is utilizing the efficient properties derived from triangular inequality based on the representative pivots and the metric index. After the high ratio filtering, only a few candidates remain and the refinement

phase examines the real distance to confirm if a candidate is an answer to q's RFN. The PIV algorithm was reported outperforming all the related ones.

However, we found that the PIV algorithm is not always efficient in any cases. Simply, issue an arbitrary query q in any location, when q is far enough from the whole dataset O, any filtering and refinement become unnecessary because it is obvious that all objects in O are answers to q's RFN. While indeed, measuring such "how far is enough" is not easy to predict. Motivated by this, we are aiming at figuring out such "far enough" boundary (say, safe area in this work), and designing a novel algorithm to further accelerate the query processing on the fly. The details of our new approach are described in the following sections.

3   Preliminaries

Since this work departures from our previous one [12], the necessary preliminaries are roughly described in this section. Firstly, we formulate the definition of reverse furthest neighbor (RFN) query as follows. Without loss of generality, we assume that the distance measure is on the 2D Euclidean space in this paper, yet indeed, it is not limited and can be on any dimensional metric spaces, and $dist(\cdot, \cdot)$ denotes the distance function.

Definition 1 (RFN). *Given an object set O and a query q, the answer to q's reverse furthest neighbor is denoted by* $\mathbf{RFN(q)} = \{o|o \in O \wedge \forall p \in O, dist(o,q) > dist(o,p)\}$.

Based on [12], the proposed PIV algorithm utilizes one theorem and two lemmas for filtering as follows. Theorem 1 essentially tells that *only if q is on the boundary of, or outside the convex hull C_O, the RFN(q) may be found*. Otherwise, no answer can be found if q inside C_O. Lemma 1 indicates if such an object o satisfies the upper bound condition (Eq. 1), o
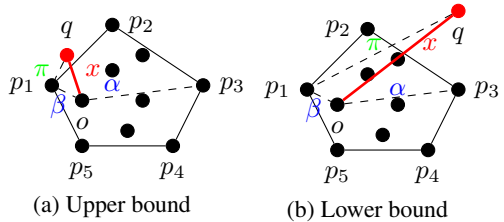
Fig. 2. (a) Lemma 1 and (b) Lemma 2

should be discarded as it has no chance to appear in the answer set of RFN(q). Oppositely, Lemma 2 guarantees such an object o that satisfies the lower bound condition (Eq. 2), safely included into RFN(q). As prerequisites of the two lemmas, the vertices of C_O are selected as a pivot set S_{piv}, pairwise distances $dist(o,p_i)$ for $o \in O$ and $p_i \in S_{piv}$ are pre-computed and stored in an index (w.r.t β in Fig. 2), and the furthest distance $max_{p_j \in S_{piv}}\{dist(o,p_j)\}$ for each o is recorded as well (i.e., α in Fig. 2). In addition, $dist(p_i,q)$ (i.e., π in Fig. 2) can be quickly computed on the fly.

Theorem 1. *Given an object set O, its convex hull C_O, and a query q, if q is inside C_O, then the answer set of RFN(q) is empty.*

Lemma 1 (Upper bound). *$\forall o \in O$, if o satisfies Eq. 1, o can be excluded from RFN(q).*

$$\exists p_i, \ dist(o,p_i)+dist(p_i,q) < \max_{p_j \in S_{piv}}\{dist(o,p_j)\}. \tag{1}$$

Lemma 2 (Lower bound). $\forall o \in O$, if o satisfies Eq. 2, o is safely included into $RFN(q)$.

$$\exists p_i, \ |dist(o, p_i) - dist(p_i, q)| > \max_{p_j \in S_{piv}} \{dist(o, p_j)\}, \tag{2}$$

4   Approach for Arbitrary Query

As we mentioned before, the PIV algorithm is not always efficient for an arbitrary query q that is without constraint of location. Therefore, in this section, we first exhibit the evidence to analyze the inefficiency of PIV algorithm. Based on the analysis, we present the details to figure out a non-trivial safe area guaranteed by strict mathematical proof. Subsequently, an adapted algorithm, named PIV^+, is designed to enhance the query processing, and its algorithmic complexity is discussed as well.

4.1   Inefficiency of PIV

For an arbitrary query q, its location can be considered in three cases. 1) When q is located inside the convex hull C_O of dataset O, the query processing terminates immediately according to Theorem 1, because it is guaranteed that there is no answer to $RFN(q)$. 2) If q is located outside C_O, filtering and refinement processing involving in algorithm PIV should be executed (e.g., in Fig. 3 w.r.t q). 3) Suppose that q is issued considerably apart from C_O, say "far enough" here, it is possible to return the whole dataset O as $RFN(q)$, as soon as q's location exceeds a certain boundary. As depicted in Fig. 3 w.r.t $q\prime$ locating "far enough", at this moment, we do not know the shape of the exact boundary. Thus, we conjecture that there would be a boundary that maybe look like an arc as imagining to be in bold (and red color) in Fig. 3. If our conjecture concerning such boundary is computable, we could efficiently answer $RFN(q)$ without filtering or refinement in this case.
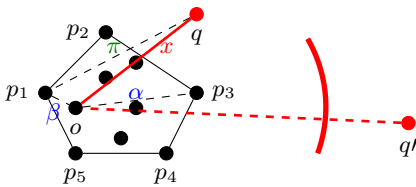
Fig. 3. Conjecture of "far enough" boundary
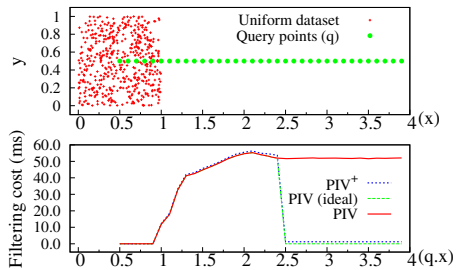
Fig. 4. Evidence of inefficiency

Anyway, the PIV algorithm must be inefficient for such a query that exceeds the boundary. So to confirm the (in)efficiency, we performed an experiment (details in Section 5). We sequentially issued different 2D queries $\{q|q.x \in [0.5, 4), q.y = 0.5\}$ to launch the PIV algorithm on a uniform dataset $O \in [0, 1]^2$ having 10,000 points. As the result demonstrated in Fig. 4, the filtering cost of PIV algorithm does not decrease even

though q is very far from O. Theoretically, when $q.x > 1 + \sqrt{2}$, the whole dataset O should be naturally returned as answer to $RFN(q)$ thus no filtering cost needs. It means that the trend of the curve should drop down asymptotic to zero when $q.x$ exceeds the threshold value, looking like the curve depicted in dotted line and labelled by "PIV (ideal)".

4.2   Construction of Safe Area

In order to identify the boundary, the necessary definitions are depicted in Fig. 5. They are diameter of convex hull (d), vertex circle ($vc(p)$), and safe area Ω, details of those are presented as follows.

Definition 2 (Diameter of convex hull). *In a dataset O, and given its convex hull C_O, the diameter of C_O is the maximal distance between any two vertices on the convex hull, which is denoted by $d = max\{dist(u,v)|u,v \in C_O\}$.*

As shown in Fig. 5, the dashed line inside the convex hull C_O w.r.t dataset O denotes the diameter d of C_O. From the definition, d is the maximal one among all the pairwise distances between any two vertices on the convex hull C_O. Meanwhile, according to the conclusion in [4], the diameter of convex hull is also the diameter of the whole dataset. It is natural to know that d is also the maximal one of the pairwise distances in dataset O. It is an important property to quickly answer $RFN(q)$. For a given query q and $\forall o \in O$, if $|oq|>d$ is satisfied, the whole dataset O can be safely included as answers to $RFN(q)$. Along this clue, we attempt to construct a safe area which guarantees that

Fig. 5. Diameter: d, Vertex circle: $vc(p)$, Safe area: $\Omega(p)$

such condition $|oq|>d$ ($\forall o$) holds. Consequently, we introduce another helpful definition, vertex circle, in the following.

Definition 3 (Vertex circle). *For a given convex hull C_O, a circle is called "vertex circle", when it is centered at a vertex p on C_O, with radius d. A vertex circle is denoted by $vc(p)$.*

The dashed circle drawn in Fig. 5 denotes the vertex circle $vc(p)$ w.r.t the specified vertex p on C_O. We then employ the vertex circle and other geometric elements to define a non-trivial safe area by its constructing steps as follows.

Definition 4 (Safe area). *Given a dataset O and its convex hull C_O, for a specified vertex p on the C_O, the safe area $\Omega(p)$ w.r.t p is constructed in the following steps.*

- *Find p's two adjacent vertices u, v on convex hull C_O.*
- *Draw vertex circle $vc(p)$ centered at the vertex p.*
- *Draw two tangent lines touching $vc(p)$, paralleling to $\overline{up}, \overline{vp}$, respectively.*
- *Draw two perpendicular bisectors of $\overline{up}, \overline{vp}$, respectively.*

The safe area $\Omega(p)$ is denoted by the intersecting semi-closure area apart from $vc(p)$.

Typically, we define the safe area of the dataset (*i.e.*, $\Omega(O)$) by taking the union of $\Omega(p)$ for $\forall p \in C_O$. It is worth noting that a specific q is only associated with the nearest vertex p of C_O, and with $\Omega(p)$ as well.

$$\Omega(O) = \bigcup_{\forall p \in C_O} \Omega(p) \tag{3}$$

For instance in Fig. 5, the semi-closure area apart from the dataset, which is separated by the (red) bold boundary, denotes the safe area $\Omega(p)$ w.r.t p. By utilizing such safe area, we have a non-trivial finding concluded in Theorem 2.

Theorem 2. *For an arbitrary query q outside C_O, if q locates inside the "safe area" $\Omega(p)$ w.r.t $p \in C_O$ (not on the boundary), then $RFN(q) = O$.*

Proof. Given q, then its nearest convex hull vertex, p, is determined. Hence q can only locate in either of the two areas in the following.

1. q apart from the arc of $vc(p)$ (Fig. 6).
2. q apart from one of the two tangent lines (Fig. 7).
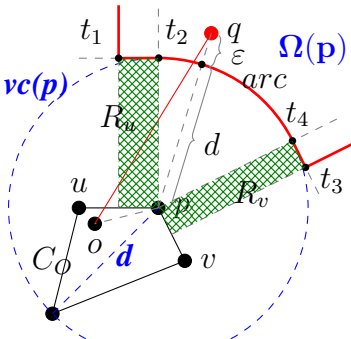
Fig. 6. Case 1: q apart from $\overset{\frown}{t_2t_4}$
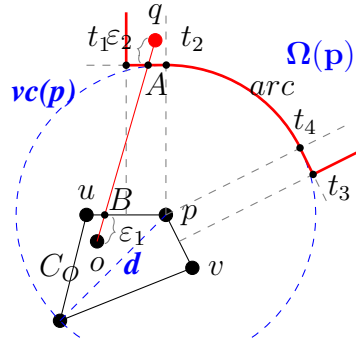
Fig. 7. Case 2: q apart from $\overline{t_1t_2}$, or $\overline{t_3t_4}$

During the proof, we import three assistant symbols, $\varepsilon, \varepsilon_1, \varepsilon_2$, in Fig. 6 and 7. In Theorem 2, "q not on the boundary", hence, $\varepsilon > 0$, and $\varepsilon_2 > 0$. And since $\forall o \in O$, $\varepsilon_1 \geq 0$.

Case 1

$$\because \overline{t_1t_2} \parallel \overline{up}, |\overline{pt_2}| = d, \overline{t_1t_2} \text{ is the tangent line of } vc(p).$$
$$\therefore \overline{t_1t_2} \perp \overline{pt_2}, \angle pt_2t_1 = \angle upt_2 = 90°.$$

Apparently, $\angle opq \geq \angle upq \geq \angle upt_2 = 90°$. This means that $\angle opq$ is the largest angle in $\triangle opq$, hence the edge $\overline{oq}$ across from it is also the largest one in the triangle. This leads to

$$|\overline{oq}| \geq |\overline{pq}| = d + \varepsilon > d \ (\varepsilon > 0).$$

Recalling that d is the diameter of O, hence for any $o \in O$, $\forall o\prime \in O$, $dist(o, o\prime) \leq d < |\overline{oq}|$, which means q is the farthest neighbor of o. This is to say that the whole dataset O becomes the answer to $RFN(q)$.

Case 2

Let $\overline{oq}$ intersects $\overline{t_1t_2}$ and $\overline{up}$ at A and B, respectively. Since the perpendicular distance between the two parallel lines $\overline{t_1t_2}$ and $\overline{up}$, is d, then $|\overline{AB}| \geq |\overline{t_2p}| = d$. Hence, we have

$$|\overline{oq}| = \varepsilon_1 + |\overline{AB}| + \varepsilon_2 > |\overline{AB}| \geq d \ (\varepsilon_1 \geq 0, \varepsilon_2 > 0).$$

Similar to the discussion in Case 1, this leads to the conclusion that all objects in O are answers to $RFN(q)$. Finally, the case when q locates apart from $\overline{t_3t_4}$, i.e., $\overline{oq}$ intersects $\overline{t_3t_4}$, is similar. □

In addition, the safe area $\Omega(p)$ w.r.t p is a semi-closure and irregular boundary, which makes it not easy to express in a simple mathematical form. Instead, for the simplicity of computation, we adopt the combination of regular shapes to examine whether q is inside $\Omega(p)$ or not. For instance, the rectangles R_u and R_v (in Fig. 6) can be computed by solving the intersections between the tangent lines t_1t_2, t_3t_4 and the circle $vc(p)$, respectively. Subsequently, the judgement can be accomplished by testing the following boolean condition, $|\overline{pq}| > d \wedge q \not\subseteq R_u \wedge q \not\subseteq R_v$. If it is true, q must be inside $\Omega(p)$, otherwise not.

4.3   The Algorithm PIV$^+$

Based on theoretical guarantee presented in Section 4.2, we design an adapted algorithm (shortly named PIV$^+$) to speed up the RFN query processing. In Algorithm 1, we can easily pre-compute the safe area $\Omega(O)$ (Eq. 3) w.r.t dataset O, by iterating all the vertices on the convex hull C_O and then taking their union. When an arbitrary query q is issued, find its nearest vertex p from C_O, *i.e.*, S_{piv} (line 1), then retrieve the safe area $\Omega(p)$ corresponding to p (line 2). If q is inside $\Omega(p)$, the algorithm returns the whole dataset O as answer to $RFN(q)$ immediately (line 4). Otherwise, call the algorithm PIV to retrieve answers (line 6). For examining $q \in \Omega(p)$, we also design a function (named InsideOmega) based on the discussions in Section 4.2.

Algorithm 1. PIV$^+$(Query q, Pivots S_{piv}, Dataset O)

Require: q, query; O, dataset; S_{piv}, convex hull of O.
   Pre-compute safe area $\Omega(O)$ iterating vertices on S_{piv}.
 1: Search p over S_{piv}, which is the nearest vertex to q;
 2: Get safe area $\Omega(p)$; { Definition 4 }
 3: **if** **InsideOmega**$(q, \Omega(p), d)$ **then** { Examine $q \in \Omega(p)$ }
 4:     **return** O; { Theorem 2 }
 5: **else**
 6:     **return**   **PIV**(q, S_{piv}, O); { Algorithm PIV [12] }
 7: **end if**

Algorithmic Complexity: Concerning the complexity of algorithm PIV$^+$, in the best case, it only costs $O(m)$ for seeking the nearest vertex to q on C_O. As the fact, $m =$

Function 1. InsideOmega(Query q, Safe area $\Omega(p)$, Diameter d)

1: $(u, v) \leftarrow$ two adjacent vertices of p;
2: Compute rectangles R_u, R_v;
3: **if** $|\overline{pq}| > d \wedge q \nsubseteq R_u \wedge q \nsubseteq R_v$ **then**
4:     **return** TRUE;
5: **else**
6:     **return** FALSE;
7: **end if**

$|C_O|$ is indeed very small even though for a dense and large dataset O, usually $m \ll N = |O|$. Although in the worst case, the complexity of PIV$^+$ reaches the same order of magnitude as PIV. Generally speaking, for an arbitrary query q on a limitless space, the more probabilities entering the safe area q has, the more efficient PIV$^+$ is than PIV. The corresponding experiment results in Fig. 9 and 10 are also positively confirming this complexity analysis.

5   Experimental Evaluations

In this section, we report the extensive results of our experimental evaluations. All the experiments are executed on an Intel-based computer and Linux OS. The CPU is Intel(R) Xeon (R) 2.83 GHz and the amount of main memory is 16.0GB. The programs are implemented in C++ language, using the open-source libraries: Spatial Index Library[1] and CGAL Library[2]. For taking into account IOs, the page size is set to 4KB.

In order to make the comparison fair, we use the same datasets as used in [12, 20]. Two kinds of datasets are used for experiments: three synthetic datasets and a real dataset in Fig. 8. The synthetic datasets include Uniform distribution (UN), Correlated Bivariate (CB), and Random-Cluster distribution (RC). The real dataset is obtained from the digital chart of the world server[3]. The real dataset (Map) is merged from 3 kinds of 2-D point data defining the road networks for California (CA) and its interest points, San Francisco (SF) and USA (US), totally 476,587 points. All the datasets are normalized to the space $\mathbb{S} = [0, L_0]^2, L_0 = 100000$. Then we randomly select different sizes of sub datasets: 10K, 50K, 100K, 200K, 300K, and 400K, as input to the programs. To guarantee an arbitrary query could appear (1) inside the convex hull C_O,
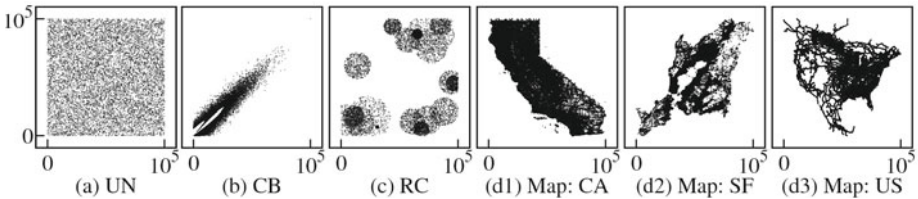
Fig. 8. Datasets: Synthetic (a), (b), (c); Real (d1), (d2), (d3)

---

(2) outside but not far from C_O, and (3) "far enough" from C_O, we randomly generate 100 queries on the space $\mathbb{S}' = [-2L_0, 3L_0]^2$ as default. All the experiment results are reported on the average of 100 queries. For the performance measurement, our proposed algorithm (PIV$^+$) is compared with the other two algorithms (PIV, and CHFC).

5.1   Effectiveness of PIV$^+$

Firstly, we reproduced the drawback of PIV algorithm. As mentioned in Section 4.1, Fig. 4, we fix a uniform distribution dataset containing 10K objects, and vary the location of q from inside of convex hull C_O to "far enough". We then run the PIV$^+$ and PIV algorithms for comparison and present the result in Fig. 4. As $q.x$ increase, the *ideal* filtering cost of PIV is expected to follow the trend of the dotted curve that is labelled in "PIV (ideal)". When $q.x < 1$, q is inside the convex hull C_O thus the algorithm needs nearly zero cost, so is the case when $q.x$ exceeds $1 + \sqrt{2}$. Otherwise, if q is within the range, the PIV algorithm needs extra cost to perform filtering. However, the curve denoting the *real* performance of PIV, is still keeping the similar trend. In contrast, the proposed PIV$^+$ algorithm is consistent with our expectation, running to the same trend of the curve "PIV (ideal)". This expressly confirms the effectiveness of the PIV$^+$ algorithm that overcame the drawback of PIV.

To respond to the complexity analysis mentioned in Section 4.3, we count the size of pivot set (m) and the size of dataset (N). The results are presented in Fig. 9, (a) referring to the absolute cardinality m versus N and (b) concerning the ratio of m/N versus N. Fig. 10a indicates that m does not have any linear nor logarithmic increase with N, and m

Fig. 9. m vs. N on different datasets

appears to be stable on different datasets. Fig. 9b further tells that m is relatively much smaller than N, and their ratios are less than 1.00% in our experiments.

Besides, we conduct the experiment to examine how the filtering cost is influenced by the probability, of which the query q enters the safe area Ω. For this purpose, we vary the range of the space $\mathbb{S}'$ where the 100 queries are randomly generated from. Let $\mathbb{S}' = [\frac{L_0 - L}{2}, \frac{L_0 + L}{2}]^2$, $L \in [0, 800000]$, and the sizes of different datasets be fixed N to 200K. By this setting, the probability of q entering Ω increases as L enlarges. As shown in Fig. 10, the results of comparing PIV$^+$ with PIV based on four different datasets exhibit similar trends of the filtering cost. It can be clearly confirmed that the filtering cost of PIV$^+$ dramatically drops down when L exceeds a certain threshold. That is where the case q entering Ω happens. Moreover, the filtering cost continues to decrease as the increase of the probability of q entering Ω, and to asymptotically trend towards the little cost $O(m)$ for seeking the nearest vertex to q on C_O that is the best case mentioned in Section 4.3.

5.2   Efficiency and Scalability of PIV$^+$

Since we have observed the influence of filtering cost by changing the query set as afore-mentioned, here we fix the query set as described at the beginning of the

Fig. 10. Filtering cost: varying query set $|Q|$, fixed N=200K

experiments. We first vary the size of dataset (N) to evaluate the efficiency of the PIV^+ algorithm only in terms of filtering cost comparing with PIV. The results on four different datasets are illustrated in Fig. 11, which implicate that the PIV^+ algorithm speeds up approximate 2 times than PIV.
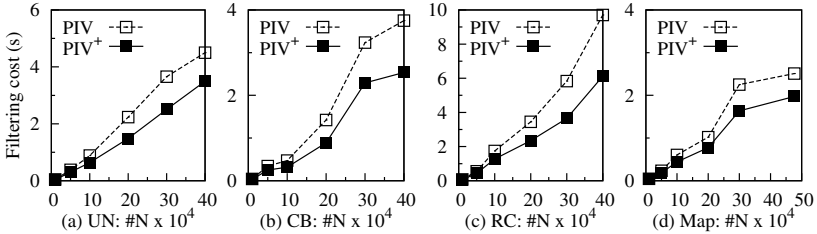
Fig. 11. Filtering cost: varying data size N on different datasets
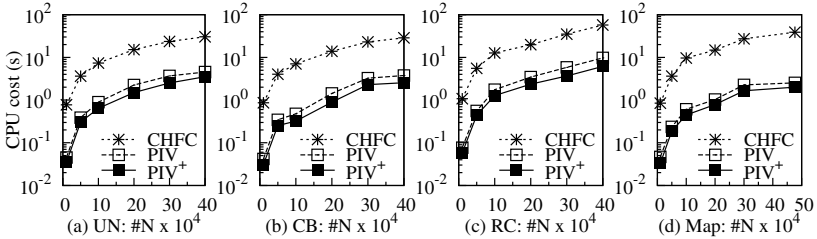
Fig. 12. Total CPU cost (CHFC vs. PIV vs. PIV^+) on different datasets
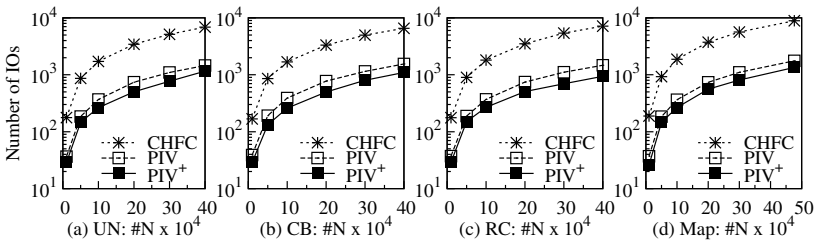
Fig. 13. Number of IOs (CHFC vs. PIV vs. PIV^+) on different datasets

To confirm the efficiency and scalability in terms of total CPU cost and number of I/Os, we additionally select the fastest algorithm CHFC in the related work as a competitor to PIV$^+$. We change different sizes of four datasets to test the efficiency and scalability using the same settings as above. As the results reported in Fig. 12 and 13, the PIV$^+$ algorithm outperforms the others in all aspects. For the total CPU cost, no matter on synthetic or real datasets, the PIV$^+$ algorithm performs the best, approximate 2 times faster than PIV and over 10 times faster than CHFC. In terms of I/O cost, the PIV$^+$ algorithm also exhibits similar better trend than the competitors. Concerning the aspect of scalability, more exactly the stability, our proposed algorithms PIV$^+$ and PIV perform better than CHFC by observing the trends of the curves in Fig. 13 which is in logarithmic scale. Approximately, our algorithms are more stable and scalable than CHFC in one order of magnitude.

6 Conclusion and Future Work

Conclusion. In this paper, we introduced our new finding of the safe area Ω that guarantees without filtering cost when query $q \in \Omega$. We also designed an efficient algorithm PIV$^+$ to adapt for arbitrary RFN query, for which the previous algorithm PIV does not always perform efficiently. The effectiveness, efficiency and scalability of our approach was confirmed by extensive experiments.

Future Work. Extending this work to process RFN queries on multi-dimensional spaces with more discussions and additional experiments must be considerable plans. To further shrink the safe area and make it optimal, and to make its shape more regular for the simplicity of geometric computation should be another directions as well. Moreover, we are planning to index the pivot set by existing methods to reduce the cost for checking the safe area, and also to develop a search system integrated with RFN queries.

References

1. Achtert, E., Böhm, C., Kröger, P., Kunath, P., Pryakhin, A., Renz, M.: Efficient reverse k-nearest neighbor search in arbitrary metric spaces. In: SIGMOD, pp. 515–526 (2006)
2. Ahn, H.K., Cheng, S.W., Cheong, O., Golin, M.J., van Oostrum, R.: Competitive facility location: the voronoi game. Theor. Comput. Sci. 310(1-3), 457–467 (2004)
3. Athitsos, V., Potamias, M., Papapetrou, P., Kollios, G.: Nearest neighbor retrieval using distance-based hashing. In: ICDE, pp. 327–336 (2008)
4. Bhattacharya, B.K., Toussaint, G.T.: Fast algorithms for computing the diameter of a finite planar set. The Visual Computer 3(6), 379–388 (1988)
5. Cabello, S., Díaz-Báñez, J.M., Langerman, S., Seara, C., Ventura, I.: Facility location problems in the plane based on reverse nearest neighbor queries. European Journal of Operational Research 202(1), 99–106 (2010)
6. Chen, H., Liu, J., Furuse, K., Yu, J.X., Ohbo, N.: Indexing the Function: An Efficient Algorithm for Multi-dimensional Search with Expensive Distance Functions. In: Huang, R., Yang, Q., Pei, J., Gama, J., Meng, X., Li, X. (eds.) ADMA 2009. LNCS, vol. 5678, pp. 67–78. Springer, Heidelberg (2009)

7. Chen, H., Liu, J., Furuse, K., Yu, J.X., Ohbo, N.: Indexing expensive functions for efficient multi-dimensional similarity search. Knowl. Inf. Syst. 27(2), 165–192 (2011)
8. Dasci, A., Laporte, G.: A continuous model for multistore competitive location. Operations Research 53(2), 263–280 (2005)
9. Hale, T.S., Moberg, C.R.: Location science research: A review. Annals OR 123(1-4), 21–35 (2003)
10. Korn, F., Muthukrishnan, S.: Influence sets based on reverse nearest neighbor queries. In: SIGMOD, pp. 201–212 (2000)
11. Lian, X., Chen, L.: Similarity search in arbitrary subspaces under lp-norm. In: ICDE, pp. 317–326 (2008)
12. Liu, J., Chen, H., Furuse, K., Kitagawa, H.: An Efficient Algorithm for Reverse Furthest Neighbors Query with Metric Index. In: Bringas, P.G., Hameurlain, A., Quirchmayr, G. (eds.) DEXA 2010, Part II. LNCS, vol. 6262, pp. 437–451. Springer, Heidelberg (2010)
13. Plastria, F.: Static competitive facility location: An overview of optimisation approaches. European Journal of Operational Research 129(3), 461–470 (2001)
14. Plastria, F., Vanhaverbeke, L.: Discrete models for competitive location with foresight. Computers & OR 35(3), 683–700 (2008)
15. Roussopoulos, N., Kelley, S., Vincent, F.: Nearest neighbor queries. In: SIGMOD, pp. 71–79 (1995)
16. Stanoi, I., Riedewald, M., Agrawal, D., Abbadi, A.E.: Discovery of influence sets in frequently updated databases. In: VLDB, pp. 99–108 (2001)
17. Tao, Y., Papadias, D., Lian, X., Xiao, X.: Multidimensional reverse *k*nn search. VLDB J. 16(3), 293–316 (2007)
18. Tao, Y., Yi, K., Sheng, C., Kalnis, P.: Quality and efficiency in high dimensional nearest neighbor search. In: SIGMOD, pp. 563–576 (2009)
19. Wu, W., Yang, F., Chan, C.Y., Tan, K.L.: Finch: evaluating reverse k-nearest-neighbor queries on location data. PVLDB 1(1), 1056–1067 (2008)
20. Yao, B., Li, F., Kumar, P.: Reverse furthest neighbors in spatial databases. In: ICDE, pp. 664–675 (2009)

Memory-Aware BWT by Segmenting Sequences to Support Subsequence Search[⋆]

Jiaying Wang, Xiaochun Yang, Bin Wang, and Huaijie Zhu

College of Information Science and Engineering,
Northeastern University, Liaoning 110819, China
`wangjiaying_85@yahoo.com.cn`

Abstract. Nowadays, Burrows-Wheeler transform (BWT) has been receiving significant attentions in academia for addressing subsequence matching problems. Although BWT is a typical technique to transform a sequence into a new sequence that is "easy to compress", it can also be extended as a kind of full text index techniques. Traditional BWT requires $n \log n + n \log \sigma$ bits to build index for a sequence with n characters, where σ is size of the alphabet. Building BWT index for a long sequence on PCs with limited memory is a great challenge. In order to solve the problem, we propose a novel variation of BWT index named S-BWT, which separates the source sequence into segments. It can reduce the memory cost to $n(\log \sigma + \log n - \log k)/k$ bits, where k is the number of segments. However, querying on each segment separately using the existing approaches has to undertake the risk of losing some significant results. In this paper, we propose two query methods based on S-BWT and guarantee to find all subsequence occurrences. Our methods can not only require small memory space, but also are faster than the state-of-art BWT backward search method for long sequence.

Keywords: BWT, subsequence matching, full text index.

1 Introduction

Recently long sequence data is growing at an exponential rate, such as web data, record data, genome data, etc. Subsequence matching on these long sequences is extremely common application in commercial business and scientific research. Burrows-Wheeler transform (BWT) is an algorithm already been applied in data compression techniques [1]. It can also be used as a full text index for providing fast substring search over text collections. The advantage of BWT index is that the compact structure can save storage space to the same size of the sequence. However, building BWT with long sequences consumes too much memory. In this paper, we solve the problem by segmenting sequences.

Challenge. The challenge of searching long sequence is the enormous memory cost while building BWT. BWT needs to be constructed from suffix array (SA). Since the SA keeps all the start positions of the source sequence, every item in SA will take $\log n$ bits space, where n is the size of the source sequence. The space cost of storing SA is $n \log n$ bits. Compared to SA, BWT saves much more space. It takes about $\log \sigma$ bits for every character, where σ is size of the alphabet. Space of $n \log \sigma$ bits will be enough to keep the whole BWT. The whole space cost of building BWT is $n \log n + n \log \sigma$ bits. We expect to search a longer sequence in limited memory, and the query process should be as fast as possible.

Contribution. In this paper, we propose a novel variation of BWT called S-BWT. We separate the source sequence into segments, and build BWT respectively on each segment. The set of all the BWTs is the S-BWT, which can get rid of storing the whole suffix array. The method can save the memory cost of constructing BWT greatly to $n(\log \sigma + \log n - \log k)/k$, where k is the number of segments. We can control the memory cost by adjusting option k. Segmenting the source sequence can also accelerate the process of building index.

We also propose two methods for query searching based on the S-BWT index. For long sequence with billion characters, our methods are better than traditional BWT method. We analyze the time complexity of our algorithms. Let the length of a sequence be n, and the number of the query occurrences be occ. The time complexity of the BWT back search algorithm is $\Theta(occ + m)$, where m is the length of the query. Searching on shorter sequence earns a speed-up rate according to principle of locality. The lower bound of time complexity for searching on S-BWT is $\Theta((occ + mk)/r)$, where k is the number of segments of the sequence, and r is the speed-up rate. Time complexity of our methods are close to the lower bound.

We conduct experiments with our method and BWT backward search method. For long sequence of hundreds and thousands MB, our method is better than BWT backward search method. Notice that our method is a generalization of BWT method, when $k = 1$, our method is the same as BWT backward search method.

Related Work. The process of computing BWT from SA is simple and fast. However, constructing SA for a long sequence is a challenging job. Much work focused on researching about constructing SA [2,3,4,5,6,7]. However, all these methods had to keep the text and suffix array in memory. When the main memory is not enough, some work in [8,9] chose to use external memory. Another work in [10] chose to sample the text to distribute the SA to small blocks. There are also some space efficient algorithms to construct compressed text indices [11,12].

The BWT backward search algorithm can provide fast substring search. It was first proposed in paper [13], and the method was further discussed in latter papers [14,15,16,17]. This method had been used to solve the sequence alignment problem for genome data [18,19]. Work in paper [20] proposed a four-stage algorithm to update the Burrows-Wheeler Transform.

We introduce all associated vocabularies and structures and the problem definitions in Section 2 and give an BWT structure on overlap segmented text and query algorithm in Section 3. BWT structure on disjoint Segmented text and query algorithm are

proposed in Sections 4. There are the experimental results and comparisons of the two methods and the original BWT method in Section 5. Finally, we summarize our conclusion in Section 6.

2   Preliminaries

Let $T[0..n-1]$ be a text, where n is the length of the text. For every $0 \leq i < n$, $T[i]$ belongs to a finite ordered alphabet Σ of size σ. To ease the process of subsequence comparison, we add another character $\$$ to the end of the text. Let $T[n] = \$$. The character $\$$ does not belong to the Σ, and it is smaller than any character of Σ.

The subsequence $T[i..j]$ is a sequence starting at i position and ending at j position, where $0 \leq i, j \leq n$. If $i > j$, $T[i..j]$ is an empty sequence. If j equals to n, we call it a suffix. If i equals to 0, we call it a prefix.

Burrows-Wheeler transform of a sequence T is denoted by $BWT(T)$. It is also a sequence with $n+1$ characters. The only difference is the order of characters. The cyclic shifts of T construct a $(n+1) \times (n+1)$ matrix M. Sorting the rows in lexicographical produces the $BWT(T)$, which is the last column of M. $BWT[T[i]]$ cyclically precedes $BWT[i]$ in T. For example, BWT of `mississippi$` is `ipssm$pissii`. The detail of BWT transform is available in the original paper.

BWT can revert to T, which is called LF-mapping. Let $LF[i] = C[BWT[i]] + r_i$, where $C[BWT[i]]$ is the number of occurrences of characters smaller than $BWT[i]$, and r_i equals to the occurrences of character $BWT[i]$ in the prefix $BWT[0..i]$. We can revert T from $T[n-1]$ to $T[0]$ by setting $t = 0$, and looping j from n-1 to 0, $T[j] = BWT[t]$, $t = LF[t]$.

The backward search algorithm searches the query backward, one step a character. It refines an interval $[l, h)$. At step i, $SA[j](l \leq j \leq h)$ is the start positions of occurrences of the query's suffix $P[i..p-1]$.

Actually, BWT is constructed from suffix array, which is an array with start position of lexicographically sorted suffixes of the text. It satisfies that $T[SA[i]..n] < T[SA[j]..n]$ if $0 \leq i < j \leq n$. The formulas transform SA to BWT is depicted in Equation 1.

$$BWT[i] = \begin{cases} T[SA[i] - 1] & \text{if } SA[i] \neq 0 \\ \$ & \text{if } SA[i] = 0 \end{cases} \tag{1}$$

Problem Definition. Given a text and a query, subsequence matching problem is to find all occurrences of the subsequence in the text. Definition 1 defines the problem more formally.

Definition 1 *Let $T[0..n-1]$ be a text, and $P[0..m-1]$ be a query. Subsequence matching problem is to find all the start positions of occurrences of P in T, i.e. $\{i \mid 0 \leq i \leq n, T[i..i+m\text{-}1] = P[0..m\text{-}1]\}$.*

This problem can be solved by BWT back search algorithm. In this paper, we take the memory cost into account. The space cost is an extra restriction. The process should guarantee the efficiency of query and memory cost at the same time.

3   Construct BWT on Overlapped Segments

In this section, we propose a BWT structure on overlapped segments of the text T and a query processing algorithm based on the proposed structure.

3.1   Decompose a Text into Overlapped Segments

Given a text T, we decompose T into a set of overlapped segments $T_1, \ldots, T_k$ and construct BWT for each segment $T_i (1 \le i \le k)$. Let $BWT(T_i)$ be the transformation from segment T_i. We use $S\text{-}BWT_O(T)$ to denote the BWT transformation for these segments $T_1, \ldots, T_k$, denoted $S\text{-}BWT_O(T) = \{BWT(T_1), \ldots, BWT(T_k)\}$.

Fig. 1 shows the sketch of decomposing a text T into overlapped segments. Let k be the number of segments and l be the overlapping length between any two adjacent segments. Let L_i be the length of segments T_i and $r = (n + l \times k - l)\%k$. The value L_i can be computed using Equation 2.

$$L_i = \begin{cases} \lceil (n + l \times k - l)/k \rceil & \text{if } 1 \le i \le r \\ \lfloor (n + l \times k - l)/k \rfloor & \text{if } r + 1 \le i \le k \end{cases} \tag{2}$$

Take mississippi as an example. Let $k = 2$ and $l = 4$, the text can be decomposed to mississi and issippi, where issi is the overlapped substring.

Fig. 1. Sketch of decomposing a text into overlapped segments

Since we work on each segment with L_i characters, the memory cost of transforming T to $S\text{-}BWT_O(T)$ is at most $(n + l + k) \times (\log \sigma + \log(n + l + k) - \log(k))/k$ bits, where σ is size of the alphabet.

3.2   Query Processing Using BWTs on Overlapped Segments

Given a query P with length m, we want to find all occurrences of P in the text T using $S\text{-}BWT_O(T)$. In this section, we assume that a query length is less than or equal to l. In Section 4 we relax this assumption. Fig. 2 shows the two cases that the query P could appear in the segments.

Case 1: An answer to the query only appears in an individual segment T_i.
Case 2: An answer to the query appears in the overlapped region of two adjacent segments T_i and T_{i+1}.

Fig. 2. Two cases of a query on overlap segments

Querying on $S\text{-}BWT_O(T)$, the order of characters in segment T_i is different from the order in T_i. It causes a problem that it is not easy to find the overlap area between T_i and T_{i+1}.

If an occurrence of a query is in the overlapped region, it must start at range $[0, l-m]$. We call the range a *filter interval*. All the occurrences starting at positions in a filter interval should be ignored, since it is already found in T_i. Fig. 3 shows the filter method. P_1 in Fig. 3(a) starts in filter interval, so it should be filtered. While P_2 in Fig. 3(b) starts out of the filter interval, which is valid and it can not appear in the overlapped region.

Fig. 3. Filter method for overlapped segments

The process of answering the query using $S\text{-}BWT_O(T)$ is listed in Algorithm 1. We search the query on each segment with procedure `backwardSearch`[13] (line 3). As we introduced in Sections 2, each step of the `backwardSearch` algorithm reports an SA interval $[l, h]$, and each SA value $SA[j](l \leq j \leq h)$ is the start positions of occurrences of the query's suffix $P[i..m-1]$, where m is the length of given query. Therefore, step m of backward search algorithm reports the interval $[l, h]$, and $SA[j](l \leq j \leq h)$ is the start positions of occurrences of the query. We then filter the results (line 5). We collect all the filtered results to get the answer (line 6).

The time complexity of Algorithm 1 is $\Theta(occ+\delta+mk)$, where δ is the query occurrences in the overlapped region. The method is fast, since searching on shorter sequence earns a speed-up rate according to the principle of locality. However, the method does not apply to the situation when the query is longer than overlapping length. In order to solve this problem, we have to rebuild $S\text{-}BWT_O(T)$ using a longer overlapping length. Obviously, it is not desirable. In Section 4, we propose another segment structure to solve this problem.

Algorithm 1: Answer query using $S\text{-}BWT_O(T)$

Input: $S\text{-}BWT_O(T)$ $sbwt[1..k]$, Pattern P, overlapping length l
Output: Start positions of all appearances of P

1   $m \leftarrow P.length$;
2   **for** $i \leftarrow 1$ **to** k **do**
3      $tmp = \texttt{backwardSearch}(sbwt[i], P)$;
4   **if** $i \neq 1$ **then**
5         $tmp \leftarrow filter(tmp)$;
6      $res \leftarrow res \cup tmp$;
7   **return** res;

---

4   Construct BWT on Disjoint Segments

In this section, we propose another BWT structure on disjoint segments of the text T and two query processing algorithms based on this proposed structure.

4.1   Decompose a Text into Disjoint Segments

Given a text T, we decompose T into a set of disjoint segments $T_1, \ldots, T_k$. We build a BWT for each segment. We utilize $S\text{-}BWT_D(T)$ to denote the BWT transformation for these segments $T_1, \ldots, T_k$, denoted $S\text{-}BWT_D(T) = \{BWT(T_1), \ldots, BWT(T_k)\}$.

$S\text{-}BWT_D(T)$ is similar to $S\text{-}BWT_O(T)$ except that the segments do not have overlap. We try to decompose T into segments of same length L', where $L' = \lfloor n/k \rfloor$. We handle the remainder the same as $S\text{-}BWT_O(T)$. The memory cost is $n(\log \sigma + \log n - \log k)/k$ bits. Take $\texttt{mississippi}$ for example. Let segmented number be 2. The text is decomposed into $\texttt{missis}$ and $\texttt{sippi}$.

4.2   Query Processing Using BWTs on Disjoint Segments

Given a query P with length m, we want to find all the occurrences of P in the text T using $S\text{-}BWT_D(T)$.

Case 1: An answer to the query only appears in an individual segment. The process can find it. This case is the same as case 1 depicted in Fig. 2.

Case 2: An answer to the query is crossing two adjacent neighbor segments. If we search answers in each $BWT(T_i)$ in $S\text{-}BWT_D(T)$ separately, it will lose answers. In this section, we focus on this case and propose two algorithms to guarantee to find all the answers.

In order to find the query with backward searching, we need to find the suffix of the query as the prefix of a segment. The rest prefix of the query needs to be verified on the left segment. As we discussed in Sections 2, each step of BWT backward search algorithm reports an interval $[l, h]$, and $SA[j] (l \leq j \leq h)$ is the start positions of occurrences of the query's suffix $P[i..m-1]$. The naive method is to check all the positions at every step. If a position is the start of the segment, we find the suffix of the query as the prefix of the segment. However, this will take $\Theta(nm)$ time. Notice that if the interval covers the position of \$, which means that the next left characters include \$. Notice that \$ is the only character cycle before the first character, since BWT is the last column of

matrix with sorted cyclic shifts of the text as introduced in Section 2. Therefore, we find the suffix of the query as the prefix of the segment. Theorem 1 explains the checking method more formally. Time complexity of this checking is $\Theta(m)$.

Theorem 1 (BWT suffix checking). *Consider matching the query $P[0..m-1]$ on segment T_i using $BWT(T_i)$. Let endPos be the position of $\$$ in $BWT(T_i)$. Step j of backward search $BWT(T_{i+1})$ with P reports an interval $[l,h]$. If the $l \leq endPos < h$, the suffix $P[m-j..m-1]$ is the prefix $T_i[0..j-1]$.*

We collect all the positions which need to be verified in a check array. The verification of the rest prefixes of the query is on the left segment.

The process of querying on $S\text{-}BWT_D(T)$ is depicted in Algorithm 2. We initialize the *checkArray* to be $\emptyset$. We get the end position (line 3). We verify each position in *checkArray* of its right segment (lines 4 – 15). When a query spans more than two segments, the prefix will be verified iteratively on its left segment (line 8). If the suffix of the query is the prefix of the segment, the position will be collected in *checkArray* (line 16). For the convenience of the verification, we loop the $S\text{-}BWT_D(T)$ from the right to left.

Algorithm 2: Answer query using $S\text{-}BWT_D(T)$

Input: $S\text{-}BWT_D(T)$ $sbwt[1..k]$, source sequence T, Pattern P
Output: Position array res

1   $checkArray \leftarrow \emptyset$;
2   **for** $i \leftarrow k$ **to** 1 **do**
3     $endPos \leftarrow pos(\$)$;
4   **foreach** *element* $j \in checkArray$ **do**
5       $t \leftarrow$ *last position of the segment in* T;
6   **while** $j >= 0$ **do**
7   **if** $t < 0$ **then**
8           $tmpArray \leftarrow tmpArray \cup j$;
9   *break*;
10   **else if** $T[t] \neq P[j]$ **then**
11          $t \leftarrow t - 1$;
12          $j \leftarrow j - 1$;
13   *break*;
14   **if** $j < 0$ **then**
15         $res \leftarrow res \cup k + 1$;
16     $checkArray \leftarrow tmpArray$;
     $tmp = bwtCheckSearch(sbwt[i], P, endPos, checkArray)$;
17     $res \leftarrow res \cup tmp$;
18 **return** res;

Algorithm 2 can search query on $S\text{-}BWT_D(T)$, and the query process is fast. However, the method wastes larger index space, since it needs the source sequence. Time complexity of the verification of a query is $\Theta(m)$. Time complexity of Algorithm 2 is $\Theta(occ + (\eta + k)m)$, where η is the number of query crossing to neighbor segments.

Instead of verifying on the source sequence using Algorithm 2, we revert the parts of the source sequence to verify the query on the fly, denoted by `backWalk`. The

process of query on $S\text{-}BWT_D(T)$ is depicted in Algorithm 2. The verification process in Algorithm 3 does not need the source sequence. We revert parts of sequence to do the verification (lines $4 - 15$).

Algorithm 3: Answer query on $BWT_{DS}(T)$ with `backWalk`

Input: $S\text{-}BWT_D(T)$ $sbwt[1..k]$, Pattern P
Output: Position array res

```
1  m ← P.length;
2  for i ← k to 1 do
3  │   checkArray ← ∅;
4  │   foreach element j ∈ checkArray do
5  │   │   t ← 0;
6  │   │   while j >= 0 do
7  │   │   │   if bwt[i][t] = $ then
8  │   │   │   │   tmpArray ← tmpArray ∪ j;
9  │   │   │   │   break;
10 │   │   │   else if bwt[i][t] ≠ P[j] then
11 │   │   │   │   break;
12 │   │   │   t ← backWalk(t);
13 │   │   │   j ← j − 1;
14 │   │   if j < 0 then
15 │   │   │   res ← res ∪ k + 1;
16 │   checkArray ← tmpArray; tmp = backwardSearch(sbwt[i], P, checkarry);
17 │   res ← res ∪ tmp;
18 return res;
```

Procedure `backWalk` (t)

```
1  i ← t / BUCKETSIZE;
2  s ← i ∗ BUCKETSIZE;
3  count ← bucket[i][bwt[t]];
4  for j ← s to t − 1 do
5  │   if bwt[j] = bwt[t] then
6  │   │   count ← count + 1;
7  return C[bwt[t]] + count;
```

Procedure `backWalk` needs to compute r_i, as depicted in Section 2. The naive method to capture the number is to count the characters one by one, but this method consumes too much time. Our backWalk method uses a bucket structure. This is trading space for time. The back walk method is depicted in Procedure `backWalk`.

Compared to Algorithm 2, this back walk will take a little more time. However, the space is saved to about $1/2$ of the size. It is trading time for space. Actually, the speed of back walk is acceptable. Time complexity of Algorithm 2 and Algorithm 3 are the same.

5   Experiments

Our algorithms were implemented as two pairs of programs: S-BWT on overlapped segments (OSBWT) and S-BWT on disjoint segments (DSBWT). Every pair of programs

contained a program `buildIndex` to build according index from a text file and another program `querySearch` to search a query from the index. We also implemented the BWT backward search algorithm for comparison.

Our programs were written in C++ and compiled by g++ with O3 option. The g++ version is 4.4.5. All the tests were running on PC with 2.93 GHz Intel Core CPU, 4 GB main memory, and Ubuntu operating system (Linux distribution).

In order to evaluate the memory cost and efficiency of our proposed methods, we have conducted extensive experiments on real datasets. We tested on two data sets:

(1) English, which is the concatenation of English text files selected from etext02 to etext05 collections of Gutenberg Project. It is available at $Pizza\&Chili$ Corpus.
(2) DNA sequence, which contains 24 different chromosomes in human reference genome. It is available at UCSC goldenPath.

We randomly selected 100 subsequences from our data set as queries, and we took the average query time as the result. For English data, the length of each query was 10. The average occurrences of each query were 1855. For genome data, the length of each query was 20, and the average occurrences of each query were 2749.

Fig. 4 compares performance of building different indices with varied segmented number. The data size is 500 MB. Fig. 4(a) shows the ratio of our indices' memory cost to BWT's for English data. Fig. 4(b) shows the case for genome data. We varied segmented number k from 2 to 4096. Our algorithms' memory cost is close to $1/k$ of BWT's, and is stable for different datasets. The memory cost of our two indices is almost the same, because the only difference of two indices is the overlapping length. With the increase of k, memory cost of OSBWT and DSBWT decreased linearly. Fig. 4(c) and Fig. 4(d) show the build time of different algorithms. With the increasing of b, time cost of building both OSBWT and DSBWT decreased stably for English data and genome data.

Table 1 compares the performance of different algorithms with varied data size. The length of overlapped region of OSBWT was still 2000. We compare the memory cost of building OSBWT and DSBWT with BWT. We change data size from 200 MB to 1000 MB. We let $k = 256$, which is the best value we got from Fig. 5. With the increase of the data size, the memory cost of OSBWT and DSBWT is stably close to $1/k$ of original BWT's, $1/256$ in this setting. When data size is 600 MB, BWT consumed 3000 MB, while OSBWT and DSBWT only consumed about 12 MB, which is only about 0.4% of the BWT's size. The memory cost of building OSBWT and DSBWT is equal. When data size lager than 600 MB, BWT fail to build for the huge memory cost.

Fig. 5 compares the query time for OSBWT and DSBWT. We can find that the query time of both algorithms decreased with the variation of k from 2 to 256. The reason is that the algorithms earned a speed-up with the decreasing of memory cost. With the further increasing of k, the query time of both algorithms increased according to our analysis of time complexity in Sections 3 and 4.

Table 2 shows the query time of BWT, OSBWT and DSBWT using different data size. The efficiency of our three algorithms is better than BWT backward search algorithm. When the data size was small, all the algorithms performed without much difference. However, when the data size was large, our algorithms performed better

(a) Memory cost on English

(b) Memory cost on genome

(c) Build time on English

(d) Build time on genome
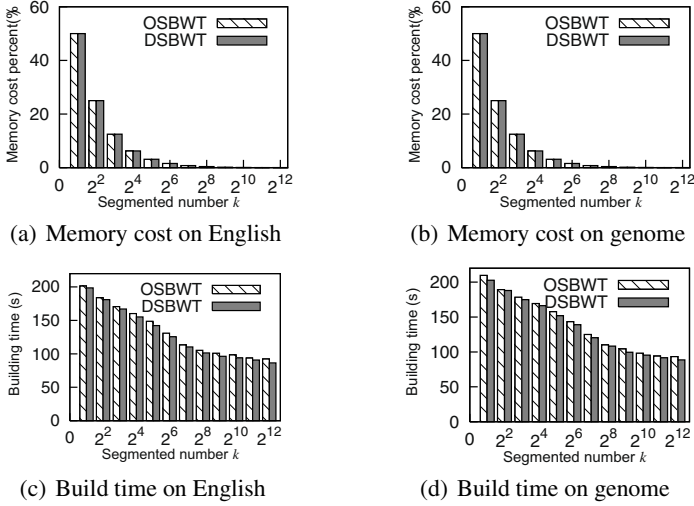
Fig. 4. Performance of building different indices with varied segmented number k

Table 1. Performance of building different indices with varied data size

performance	Data size	200MB	400MB	600MB	800MB	1000MB
english	BWT(MB)	1000.26	2000.26	3000.26	——	——
	OSBWT(MB)	4.18	8.09	11.99	15.9	19.81
	DSBWT(MB)	4.17	8.08	11.98	15.89	19.8
genome	BWT(MB)	1000.26	2000.26	3000.26	——	——
	OSBWT(MB)	4.18	8.09	11.99	15.9	19.81
	DSBWT(MB)	4.17	8.08	11.98	15.89	19.8

(a) Query time on English

(b) Query time on genome
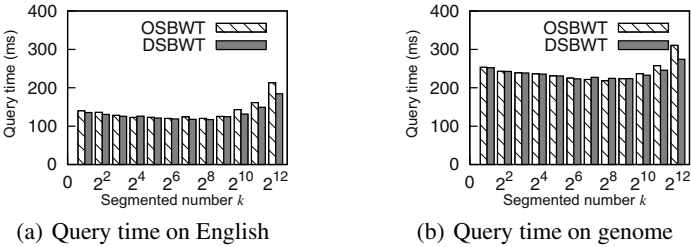
Fig. 5. Performance of different algorithms with varied segmented number k

than BWT. For larger data size, OSBWT is the fastest one according to analysis of time complexity in Section 3 and 4.

Fig. 6 shows the query time for OSBWT with varied overlapping length. With increasing of overlapping length, the query time increased. OSBWT with larger segmented number increased faster.

Table 2. Performance of different algorithms with varied data size

performance	Data size	200MB	400MB	600MB	800MB	1000MB
english	BWT(ms)	45.61	93.06	153.3	——	——
	OSBWT(ms)	48.09	96.04	140.96	187.22	269.52
	DSBWT(ms)	41.83	93.31	149.45	192.41	273.66
genome	BWT(ms)	94.53	206.82	325.18	——	——
	OSBWT(ms)	89.05	192.08	278.01	401.33	503.28
	DSBWT(ms)	47.22	178.4	293.7	424.24	529.82

(a) English data   (b) Genome data
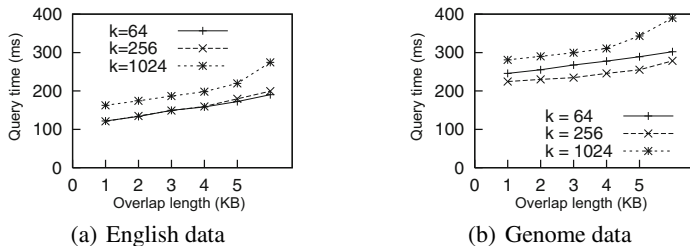
Fig. 6. Performance of OSBWT with varied overlap size

6   Conclusion

In this paper, we propose a novel variation of BWT, called S-BWT. Compared to the original BWT, S-BWT saves much more memory. Based on the S-BWT, we propose two methods for searching query on S-BWT.

Searching on shorter sequence earns a speed-up rate according to principle of locality. The time complexity lower bound of S-BWT is $\Theta((occ + mk)/r)$, where k is the segmented number, and r is the speed-up rate. Time complexity of our methods is close to the bound. Time complexity of the algorithm on $S\text{-}BWT_O(T)$ is $\Theta(occ + \delta + mk)$, where δ is query occurrences in the overlapped region. Time complexity of the algorithm on $S\text{-}BWT_D(T)$ is $\Theta(occ + (\eta + k)m)$, where η is the number of query crossing to neighbor segments.

We conduct experiments with our method and original BWT method. Compared to original BWT method, our method can build index with less memory and less time. Our query algorithms are faster than the original BWT backward search algorithm on large text of hundreds and thousands MB.

Our method can be used for searching subsequence in large text collections, which are common in the field of information retrieval in web data, record data and genome data, etc. We will combine our S-BWT with compression techniques to increasingly reduce the space cost of the index. Our method has good scalability. It can be easily expanded to work under parallel computing environment, which will further improve the performance.

References

1. Burrows, M., Wheeler, D.J.: A block-sorting lossless data compression algorithm. Technical report, SRC Research Report 124 (1994)
2. Puglisi, S.J., Smyth, W.F., Turpin, A.: A taxonomy of suffix array construction algorithms. ACM Comput. Surv. 39(2) (2007)
3. Kim, D.K., Sim, J.S., Park, H., Park, K.: Constructing suffix arrays in linear time. J. Discrete Algorithms 3(2-4), 126–142 (2005)
4. Ko, P., Aluru, S.: Space efficient linear time construction of suffix arrays. J. Discrete Algorithms 3(2-4), 143–156 (2005)
5. Manzini, G., Ferragina, P.: Engineering a lightweight suffix array construction algorithm. Algorithmica 40(1), 33–50 (2004)
6. Burkhardt, S., Kärkkäinen, J.: Fast Lightweight Suffix Array Construction and Checking. In: Baeza-Yates, R., Chávez, E., Crochemore, M. (eds.) CPM 2003. LNCS, vol. 2676, pp. 55–69. Springer, Heidelberg (2003)
7. Kärkkäinen, J., Sanders, P., Burkhardt, S.: Linear work suffix array construction. J. ACM 53(6), 918–936 (2006)
8. Crauser, A., Ferragina, P.: A theoretical and experimental study on the construction of suffix arrays in external memory. Algorithmica 32(1), 1–35 (2002)
9. Dementiev, R., Kärkkäinen, J., Mehnert, J., Sanders, P.: Better external memory suffix array construction. ACM Journal of Experimental Algorithmics, 12 (2008)
10. Kärkkäinen, J.: Fast bwt in small space by blockwise suffix sorting. Theor. Comput. Sci. 387(3), 249–257 (2007)
11. Hon, W.-K., Sadakane, K., Sung, W.-K.: Breaking a time-and-space barrier in constructing full-text indices. SIAM J. Comput. 38(6), 2162–2178 (2009)
12. Lam, T.-W., Sadakane, K., Sung, W.-K., Yiu, S.-M.: A Space and Time Efficient Algorithm for Constructing Compressed Suffix Arrays. In: Ibarra, O.H., Zhang, L. (eds.) COCOON 2002. LNCS, vol. 2387, pp. 401–410. Springer, Heidelberg (2002)
13. Ferragina, P., Manzini, G.: Opportunistic data structures with applications. In: FOCS, pp. 390–398 (2000)
14. Ferragina, P., Manzini, G.: Indexing compressed text. J. ACM 52(4), 552–581 (2005)
15. Ferragina, P., Manzini, G., Mäkinen, V., Navarro, G.: Compressed representations of sequences and full-text indexes. ACM Transactions on Algorithms 3(2) (2007)
16. Navarro, G., Mäkinen, V.: Compressed full-text indexes. ACM Comput. Surv. 39(1) (2007)
17. Sirén, J., Välimäki, N., Mäkinen, V., Navarro, G.: Run-Length Compressed Indexes Are Superior for Highly Repetitive Sequence Collections. In: Amir, A., Turpin, A., Moffat, A. (eds.) SPIRE 2008. LNCS, vol. 5280, pp. 164–175. Springer, Heidelberg (2008)
18. Li, H., Durbin, R.: Fast and accurate short read alignment with burrows-wheeler transform. Bioinformatics 25(14), 1754–1760 (2009)
19. Li, R., Yu, C., Li, Y., Lam, T.W., Yiu, S.-M., Kristiansen, K., Wang, J.: Soap2: an improved ultrafast tool for short read alignment. Bioinformatics 25(15), 1966–1967 (2009)
20. Salson, M., Lecroq, T., Léonard, M., Mouchard, L.: A four-stage algorithm for updating a burrows-wheeler transform. Theor. Comput. Sci. 410(43), 4350–4359 (2009)

Mining Frequent Association Tag Sequences for Clustering XML Documents

Lijun Zhang, Zhanhuai Li, Qun Chen, Xia Li, Ning Li, and Ying Lou

School of Computer Science and Technology,
Northwestern Polytechnical University, Xi'an 710072, China
{zhanglijun,lizhh,chenbenben,lixia,lining}@nwpu.edu.cn
louying@mail.nwpu.edu.cn
http://www.nwpu.edu.cn/jsj

Abstract. Many XML document clustering algorithms need to compute similarity among documents. Due to its semi-structured characteristic, exploiting the structure information for computing structural similarity is a crucial issue in XML similarity computation. Some path based approaches model the structure as path set and use the path set to compute structural similarity. One of the defects of these approaches is that they ignore the relationship between paths. In this paper, we propose the conception of *Frequent Association Tag Sequences (FATS)*. Based on this conception, we devise an algorithm named *FATSMiner* for mining FATS and model the structure of XML documents as FATS set, and introduce a method for computing structural similarity using FATS. Because FATS implies the ancestor-descendant and sibling relationships among elements, this approach can better represent the structure of XML documents. Our experimental results on real datasets show that this approach is more effective than the other path based approaches.

Keywords: Structural Similarity, Frequent Association Tag Sequence, Frequent Pattern, Sequential Pattern, XML Document Clustering.

1 Introduction

XML has been widely used as de facto standard for data representation and exchange. The increasing XML documents have raised some issues on how to store, filter and retrieve XML documents more effectively and efficiently. The clustering technique can facilitate such applications by grouping XML documents according to their similarity. So XML similarity computation has become a fundamental problem of many XML process technique, and applied to many fields, such as semi-structured data integration, classification/clustering of XML documents, XML retrieval, and so on [1].

Since hierarchical structure is incorporated in the XML documents, clustering of XML data significantly differs from flat data. Exploiting the structure information for computing structural similarity is a crucial issue in XML similarity computation. Some path based methods [2,3,4] represent XML documents as
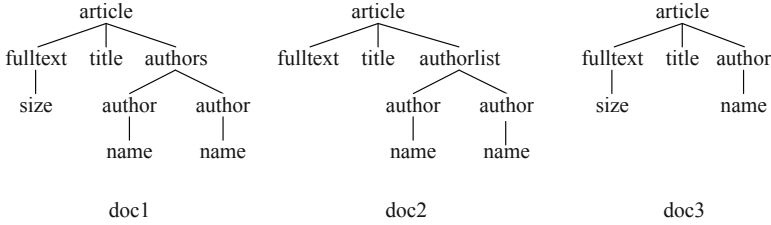
Fig. 1. XML document examples

set of paths, and then compute similarity using set or vector operations. For example, considering the three XML documents in Fig.1. In the bag of tree paths model (referred as *BOTP* in this paper) proposed in [2], a document is represented by the set of paths from the root node to the leaf nodes of corresponding DOM tree. The three documents in Fig.1 are represented as the second column in Table 1. The two pairs of paths: "article/fulltext/size" vs "article/fulltext", "article/authors/author/name" vs "article/authorlist/author/ name" in *doc1* and *doc2* are deemed different completely. In fact, the two pairs of paths are very similar. Bag of tree paths model retains information on all parent-child relationships, but it ignores the sibling relationships, for example, paths "article/fulltext/size" and "article/title" in *doc1* and *doc3* are independent in *BOTP*. A stronger model called bag of XPaths model (referred as *BOXP* in this paper) is also proposed in [2]. *BOXP* incorporates some, but not all, sibling information.

Table 1. Various path representations of XML documents

Doc	BOTP	commonXPath	subPath
doc1	article/fulltext/size, article/title, article/authors/author /name	article/fulltext/size, article/title, article/*/author/name	article/fulltext/size, article/title, article/authors/author/name, article/fulltext, article/authors, fulltext/size, authors/author, author/name, . . .
doc2	article/fulltext, article/title, article/authorlist /author/name	article/fulltext, article/title, article/*/author/name	article/fulltext, article/title, article/authorlist/author/name, article/authorlist, authorlist/author, author/name, . . .
doc3	article/fulltext/size, article/title, article/author/name	article/fulltext/size, article/title	article/fulltext/size, article/title, article/author/name, article/fulltext, article/author, fulltext/size, author/name, . . .

Reference [3] mine frequent XPaths in document collection and represent an XML document by a vector consists of frequent XPaths (called commonXPath).

Let minimum support is 60%, the documents in Fig.1 can be represented as the third column in Table 1. Though the paths "article/*/author/name" in *doc1* and *doc2* is deemed as similar, the path "article/author/name" in *doc3* is deemed different. Essentially, the paths "article/authors/author/name" , "article/authorlist/author/name" and "article/author/name" are very similar. In addition, [3] also assume the paths are independent in computing similarity by vector. For example, paths "article/fulltext" and "article/title" are all contained in the three documents, and there is sibling relationship between them, but [3] ignores the information.

Except root paths from root node to leaf node, reference [4] take all subpaths of root paths into consideration. An XML document can be represented as a set of all root paths and subpaths, as the forth column in Table 1. This method still can not solve the problem completely that paths "article/authors/author/name" and "article/authorlist/author/name" are similar , and it still ignores the sibling relationships between elements in computing similarity.

From the above example we can see, the existing path based approaches for computing structural similarity among XML documents have the following two problems:

1) It can not be handled well in the case of matching partially between paths. That is, the similarity among paths "article/authors/author/name", "article/authorlist/author/name" and "article/author/name" can not be handled well.
2) Though the parent-child/ancestor-descendant relationships between elements are captured, the sibling relationships are ignored completely or partially. That is, the paths "article/fulltext" and "article/title" in Fig.1 are independent.

In this paper, adopting the concepts of frequent and sequential pattern, we propose an approach based on frequent association tag sequences to address the two issues. Our contributions include:

1) We introduce the concept of *Frequent Association Tag Sequences (FATS)* to model the structures of XML documents.
2) We devise an algorithm named *FATSMiner* to mine the frequent association tag sequences in XML document collection.
3) We have conducted an extensive experimental study on two datasets to verify the effectiveness of the FATS approach. The results demonstrate that FATS performs better than existing path based approaches on clustering XML documents.

The rest of this paper is organized as follows. In Sect. 2 we briefly review some related work in the area of measuring structural similarity among XML documents. In Sect. 3, we describe the concept of *FATS* firstly, and then we present the algorithm *FATSMiner* and the structural similarity measurement based on *FATS*. In Sect. 4 we illustrate the effectiveness of the FATS approach in XML document clustering. Finally, in Sect. 5, we conclude the paper.

2   Related Work

Measuring the structural similarity among XML documents has been an active area of research in the past few years, and various methods have been proposed to solve the problem in the literature. As described in previous section, some methods view XML documents as set of paths[2,3,4]. Joshi et al. propose a bag of tree paths model, in which a path is a sequence from root node to leaf node[2]. Bag of tree paths model retains information on all parent-child relationships, but ignores sibling relationships. Joshi et al. extend the tree paths model, and consider XML document as a bag of XPaths[2]. XPaths incorporate some, but not all, sibling information between elements. Leung et al. represent XML document as a set of simple XPath expressions and then the structural similarity can be computed by determining the number of paths and their level of hierarchy that are similar[3]. Davood et al. represent XML document as set of all root paths and their subpaths, and then Jaccard Coefficient etc. metrics are used to compute the structural similarity[4]. However, the path based approaches have the two defects described in previous section.

Some other methods adopt tree edit distance to measure the structural similarity among XML documents[5,6]. In these methods, XML documents are represented as ordered/unordered labelled tree and it is designed to find the cheapest edit operation sequence which can transform an XML tree to another, then the sequence is used to compute the structural similarity. The differences among the various tree distance algorithms are edit operations allowed and the way of using XML tree nodes. Nierman et al. define five operations: relabel, insert, delete, insert tree, and delete tree. And they take into account XML issues such as repetitive and optional fields[5]. Zhou et al. propose an approach named MED(Merge-Edit-Distance). Given two XML document trees A and B, it compresses the two trees into one merge tree C and then transforms the tree C to the common tree of A and B with the defined operations such as delete, reduce and combine. The cost of the operation sequence is defined as Merge-Edit-Distance[6]. However, it is well known that any edit distance measure critically depends on the costs of the underlying edit operations and it has been shown by Zhang et al. that the tree edit distance computing among unordered labelled trees is NP-Complete[7].

Some additional other methods mine frequent subtrees from XML collection using data mining technique[8,9,10], and view frequent subtrees as features of XML documents. Termier et al. and Zaki propose two mining frequent subtrees methods from semi-structured data named TreeFinder[8] and TreeMiner[9] respectively, and the mined frequent subtrees can be used to compute the structural similarity. Miyahara et al. propose a method for finding all maximally frequent tag tree patterns in semi-structured data. Though frequent subtree can retain the structure information of XML documents well, most of frequent subtree mining methods are not efficient and computationally expensive, especially in the case of large dataset and the structures of documents are complex.

3   Frequent Association Tag Sequences

In this section, we first introduce the basic definitions used in this paper, include tag sequence, frequent tag sequence, association tag sequence and frequent association tag sequence etc. We then describe the frequent association tag sequence mining algorithm *FATSMiner*. Finally, we define the structrual similarity measure based on frequent association tag sequence.

3.1   Basic Definitions

An XML document can be modelled as a node-labelled directed tree where each node in the tree represents an element in the corresponding XML document. The node is labelled with the tag name of the element. We do not distinguish between elements and attributes, so attributes are treated as elements. Each edge of the tree represents a hierarchical inclusion relationship between either two elements or an element and an attribute. Since we are only interested in the structure of an XML document, we ignore text nodes. So an XML document can be represented as a set of paths from the root node to the leaf nodes. Each path from the root node to a leaf node contains the root node, the leaf node, and all the intermediate nodes required to reach the leaf node in sequence and each path can be represented by the label (tag name) sequence. Given a document collection, we can extract all tag names contained in the documents. To make our description more concise, we refer to tag name as **tag** and the set of all tags as **tagset**.

Definition 1. (Tag sequence) *A tag sequence is an ordered list of tag. We denote a tag sequence α by $< a_1, a_2, \cdots, a_n >$, where a_i is a tag in tagset. The* **length** *of tag sequence α is the number of tags contained in α. A tag sequence with length l is called an* **l-tag sequence**.

Definition 2. (Tag sequence contain relation) *A tag sequence $\beta :< b_1, b_2, \cdots, b_n >$* **contains** *another tag sequence $\alpha :< a_1, a_2, \cdots, a_m >$, if there exist integers $i_1 < i_2 < \cdots < i_m$ such that $a_1 = b_{i_1}, a_2 = b_{i_2}, \cdots, a_m = b_{i_m}$. And we say tag sequence α is a* **sub-tag sequence** *of β or tag sequence β is a* **super-tag sequence** *of α, denoted as $\alpha \subseteq \beta$ or $\beta \supseteq \alpha$. If tag sequence α is a sub-tag sequence of β, and m, the length of α, is not equal to n, the length of β, then we say tag sequence α is a* **proper sub-tag sequence** *of β or tag sequence β is a* **proper super-tag sequence** *of α, denoted as $\alpha \subset \beta$ or $\beta \supset \alpha$.*

For example, for the given documents in Fig.1, the tagset is {*article, fulltext, size, title, authors, author, name, authorlist*}. $<article, authors, author, name>$ and $<article, author, name>$ are two tag sequences, and tag sequence $<article, authors, author, name>$ contains $<article, author, name>$. Tag sequences $<article, authors, author, name>$ and $<article, author, name>$ are sub-tag sequences of $<article, authors, author, name>$. And tag sequence $<article, author, name>$ is a proper sub-tag sequence of $<article, authors, author, name>$.

A path in XML document tree can be viewd as a tag sequence and an XML document can be represented as a set of tag sequences. So, for given XML document collection, we can transform it to a tag sequence database, **TSDB**, which is a set of tuples, $<docid, tsset>$, where *docid* is the id of document and *tsset* is the set of tag sequences of corresponding document. For example, the three documents in Fig.1 consist a tag sequence database as Table 2.

Table 2. Tag sequence database

Doc	Tag Sequence Set
doc1	<article,fulltext,size>,<article,title>,<article,authors,author,name>
doc2	<article,fulltext>,<article,title>,<article,authorlist,author,name>
doc3	<article,fulltext,size>,<article,title>,<article,author,name>

Definition 3. (Support of tag sequence) *A document **supports** a tag sequence α if there exists a tag sequence β in the document, such that $\alpha \subseteq \beta$. And the **support** of tag sequence α in tag sequence database TSDB is ratio of the documents number in the database that support α to number of all documents, denoted as support(α).*

Definition 4. (Frequent tag sequence) *Given a real number $\delta(0 \leq \delta \leq 1)$ as the **minimum support threshold**, tag sequence α is **frequent** in tag sequence database TSDB if support(α) $\geq \delta$. Tag sequence α is **maximal frequent** in database TSDB, if α is frequent, and there exists no tag sequence β in TSDB, such that $\alpha \subset \beta$ and support(β) $\geq \delta$.*

Though the tag sequence retain the parent-child/ancestor-descendant relationships incorporated in XML documents structures, the sibling relationships are still ignored. Therefore, we propose the concept of association tag sequence to capture sibling information contained in XML documents.

Definition 5. (Association tag sequence) *An **association tag sequence** is a set of tag sequence, and for any tag sequence α in the set, there exists no tag sequence β in the set, such that $\alpha \subset \beta$ or $\alpha \supset \beta$. The **size** of an association tag sequence is the number of tag sequence contained in the set. An association tag sequence with size m is called an **m-association tag sequence**.*

Same to the definition of set, we can define the super and sub association tag sequence, and we omit the detailed description due to space constraints here.

Definition 6. (Support of association tag sequence) *A document **supports** an association tag sequence γ if for any tag sequence α in γ, there exists a tag sequence β in the document, such that $\alpha \subseteq \beta$. The **support** of an association tag sequence γ in a tag sequence database TSDB is ratio of the documents number in the database that support γ to number of all documents, denoted as support(γ).*

Definition 7. (Frequent association tag sequence, FATS) *Given a real number $\delta(0 \leq \delta \leq 1)$ as the **minimum support threshold**, an association tag sequence γ is **frequent** in tag sequence database $TSDB$, if $support(\gamma) \geq \delta$. An association tag sequence γ is **closed frequent** in database $TSDB$, if γ is frequent, and there exists no association tag sequence η in $TSDB$, such that $\gamma \subset \eta$ and $support(\gamma) = support(\eta)$.*

For tag sequence database $TSDB$ in Table 2, let minimum support is 100%, then tag sequences $<article, fulltext>$, $<article, title>$ and $<article, author, name>$ are frequent, and all are maximal. $\{<article, fulltext>, <article, title>\}$ and $\{<article, fulltext>, <article, title>, <article, author, name>\}$ are frequent association tag sequences. $\{<article, fulltext>, <article, title>, <article, author, name>\}$ is closed, but $\{<article, fulltext>, <article, title>\}$ is not closed.

For given XML document collection, we can mine FATSs for modelling XML documents. The adoption of sequence can solve the first problem addressed in Sect.1, and the adoption of association can solve the second problem to some extent. For example, if we use maximal frequent tag sequence $<article, author, name>$ as a feature, the documents in Fig.1 are all matched to it, it means that they are similar. If we use FATS $\{<article, fulltext>, <article, title>, <article, author, name>\}$ as a feature, then the three isolated paths are considered as a whole, and the sibling information among them are captured by the feature. In fact, a FATS exemplifies a frequent substructure: subtree or subgraph. It is a subtree in this example, as Fig.2.
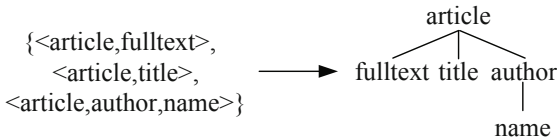
Fig. 2. FATS exemplifies frequent substructure

3.2   Mining Frequent Association Tag Sequences

Given XML document collection C, based on the definitions in 3.1, we can discover all FATSs. Then the FATSs can be used as features and each XML document can be represented by a set of FATSs. Based on the set of FATSs, we can measure structural similarity among XML documents. In order to reduce the size of feature space, we just mine closed FATSs to represent XML documents. So, if no special instructions, FATS later in this paper refer to closed frequent association tag sequence.

We devise an algorithm named *FATSMiner* for mining FATS. Fig. 3 shows pseudo-code for the *FATSMiner* algorithm. In Fig.3, line 1 is a preprocess step, and it parses the document collection C to tag sequence database, $TSDB$. The structure of each document in collection C is transformed into a set of paths which start with root node and end with leaf node, and the same paths occur only

```
Algorithm:  FATSMiner
Input:    C, XML document collection;
              δ, minimum support threshold.
Output:  FATSset, closed frequent association tag sequence set in C.

1:     TSDB=parseXML(C);
2:     FTSset = prefixspan(TSDB, δ);
3:     for (k=m;k>1;k--)  //m is the max length of tag sequences in FTSset
4:         for each k-tag sequence ts_k in FTSset
5:             delete all sub-tag sequences of ts_k from FTSset;
6:         end for
7:     end for
8:     TSDB'=convertDB(TSDB, FTSset);
9:     FATSset = closet(TSDB', δ);
10:    return FATSset;
```

Fig. 3. FATSMiner algorithm

once in the set. The attributes of the elements are processed as sub-elements. Prefixspan algorithm [11] is used to mine all frequent tag sequences (line 2). Line 3-7 finds the maximal frequent tag sequences among the set of frequent tag sequence. Then the tag sequence database is converted to $TSDB'$ (line 8), in which each document is represented by maximal frequent tag sequences only. Line 9 calls $CLOSET+$ algorithm[12] to mine all *closed FATS* with minimum support δ. Finally, the closed frequent association tag sequences set, *FATSset*, is returned (line 10).

3.3   Structural Similarity between XML Documents

We can mine all closed frequent association tag sequences, *FATSset*, from XML document collection C using the *FATSMiner* algorithm introduced in previous section. Then any XML document d_i can be represented by a set of *FATS* contained in it, denote as d_FATS_i:

$$d\_FATS_i = \{fats | fats \in FATSset \wedge d_i \ supports \ fats \wedge$$
$$\nexists fats' \in FATSset, fats' \supset fats \wedge d_i \ supports \ fats'\}$$

And then we define sets I, U and P_j^i as follows:

$$I = d\_FATS_i \cap d\_FATS_j$$

$$U = d\_FATS_i \cup d\_FATS_j$$

$$P_j^i = \{fats | fats \in d\_FATS_i \wedge fats \notin d\_FATS_j \wedge$$
$$\exists fats' \in d\_FATS_j, fats \subset fats'\}$$

Thus we define the structural similarity between XML documents as:

$$sim(d_i, d_j) = \frac{|I| + |P_j^i| + |P_i^j|}{|U|}$$

4   Experimental Study

4.1   Experimental Setting

The goal of our experiment is to examine the effectiveness of our FATS approach. For this purpose, we compare the clustering precision and recall of our method with several other path based methods. These include Joshi's method($BOTP$ and $BOXP$)[2], Leung's method($commonXPath$)[3] and Rafiei's method($subPath$)[4]. Our experiment is implemented by Java via eclipse. In the implementation of our method, we adopt $CLOSET+$ binary code in [12] and java package of prefixspan algorithm provided in [13].

Our comparison is based on two real data sets. The first, referred as Texas, is a data set introduced in [14]. This data set is generated in the XML/XSLT version of web pages from 20 different sites belonging to 4 different categories: automobile, movie, software and news&reference. There are a total of 101 documents. The second, referred as Sigmod, is the online XML version of the ACM Sigmod Record from March 1999, November 2002 and July 2005[15]. This collection contained XML files shared among three DTDs: ProceedingsPage, IndexTermsPage and OrdinaryIssuePage. We use all 33 documents from ProceedingsPage, all 94 documents from OrdinaryIssuePage, and select 100 documents from Index-TermsPage randomly. And to make the data more in line with the characteristics of heterogeneous data, we altered the DTD files a little.

4.2   Clustering Results

We cluster the two XML datasets using the method proposed in this paper and the other four methods. Same to [3], we take use of the agglomerative hierarchical clustering algorithm[16], which starts by placing each document in its own cluster and then merges these atomic clusters into larger and larger clusters, until the cluster number condition is satisfied. And we evaluate the clustering results by macro average precision and recall (refer to [17] for details). We get different results by different minimum support thresholds. The best results are acquired when the minimum support thresholds are 10% and 30% on Texas and Sigmod dataset respectively. The results are presented in Fig.4.

From the figures, we can see that on Texas dataset, the $BOTP$ method was found to be the worst, and $commonXPath$ and $FATS$ methods outperforms other methods, $FATS$ method performs better a little than $commonXPath$. On Sigmod dataset, the performance of $BOTP$, $BOXP$ and $subPath$ methods are nearly the same. $FATS$ and $commonXPath$ are significantly better than the other methods. And $FATS$ outperforms $commonXPath$ approach by 18.6% and 5.4% in precision and recall.

4.3   Effect of Minimum Support Threshold

We study the effect of minimum support on number of frequent tag sequences and clustering precision. Because $commonXPath$ method is effected by minimum
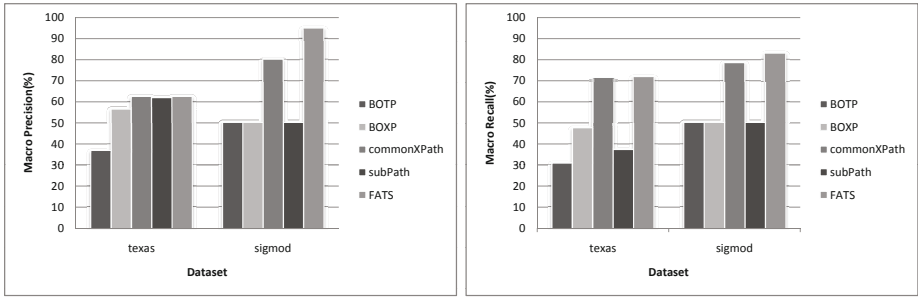
(a) Precision

(b) Recall

Fig. 4. Clustering result of the five approaches
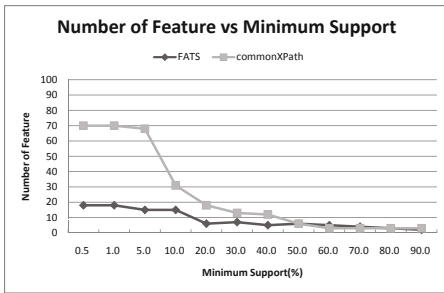
Fig. 5. Number of features with different minimum support
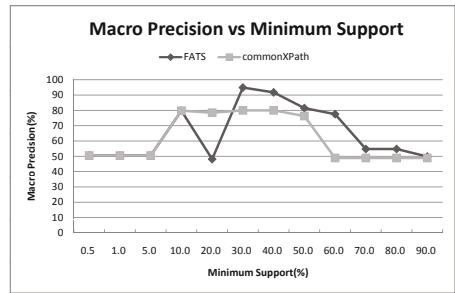
Fig. 6. Precision of clustering with different minimum support

support also, we compare it with *FATS*. The results based on Sigmod dataset are shown in Fig.5 and Fig.6.

In Fig.5, the number of features decreases as the minimum support threshold increases. When the minimum support threshold is higher than 50%, the number of FATS is nearly equal to that of commonXPath. For minimum support threshold between 20% and 50%, the number of commonXPath is greater than FATS. And for minimum support threshold below 20%, the number of commonXPath increases rapidly than FATS. That means *FATS* approach represents the XML documents with fewer features, and this leads to decreasing computation cost of distance matrix for the clustering step. So *FATS* approach is more efficient than *commonXPath* in clustering.

The precision of *FATS* and *commonXPath* for different minimum support threshold on Sigmod dataset is reported in Fig.6. We observed that the precision is the highest when the minimum support is 30% both in *FATS* and *commonXPath* approaches. When the minimum support is below 10%, the precision of *FATS* and *commonXPath* is nearly the same. When the minimum support is higher than 30%, the precision of *FATS* is higher than *commonXPath* and decreases with the increase of support.

5   Conclusions

Owing to XML documents include structure information, how to measuring structural similarity has become a crucial issue in database and information retrieval communities. There exist some previous works on this issue and some methods based on path ignore the hierarchical and/or sibling information. In this paper, we propose the concept and mining algorithm of frequent association tag sequence for computing structural similarity between XML documents and apply it to cluster XML documents. The XML documents can be represented by FATS, which can reflect the ancestor-descendant and sibling relationships between elements and compensate the shortcoming of the path based methods. As demonstrated by the experimental results, the proposed approach is more effective than the other path based approaches.

References

1. Tekli, J., Chbeir, R., Yetongnon, K.: An overview on XML similarity: Background, current trends and future directions. Computer Science Review 3(3), 151–173 (2009)
2. Joshi, S., Agrawal, N., Krishnapuram, R., Negi, S.: A Bag of Paths Model for Measuring Structural Similarity in Web Documents. In: Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 577–582 (2003)
3. Leung, H.P., Chung, F.L., Chan, S.C., Luk, R.: XML Document Clustering Using Common XPath. In: Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration, pp. 91–96 (2005)
4. Rafiei, D., Moise, D.L., Sun, D.: Finding Syntactic Similarities Between XML Documents. In: Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA), pp. 512–516 (2006)
5. Nierman, A., Jagadish, H.V.: Evaluating Structural Similarity in XML Documents. In: Proceedings of the 5th International Workshop on the Web and Databases (WebDB), pp. 61–66 (2002)
6. Zhou, C., Lu, Y., Zou, L., Hu, R.: Evaluate Structure Similarity in XML Documents with Merge-Edit-Distance. In: Washio, T., Zhou, Z.-H., Huang, J.Z., Hu, X., Li, J., Xie, C., He, J., Zou, D., Li, K.-C., Freire, M.M. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4819, pp. 301–311. Springer, Heidelberg (2007)
7. Zhang, K., Statman, R., Shasha, D.: On the Editing Distance Between Unordered Labeled Trees. Information Processing Letters 42(3), 133–139 (1992)
8. Termier, A., Rousset, M.C., Sebag, M.: TreeFinder: a First Step towards XML Data Mining. In: Proceedings of IEEE International Conference on Data Mining (ICDM), pp. 450–457 (2002)

9. Zaki, M.J.: Efficiently Mining Frequent Trees in a Forest: Algorithms and Applications. IEEE Transactions on Knowledge and Data Engineering 17(8), 1021–1035 (2005)
10. Miyahara, T., Suzuki, Y., Shoudai, T., Uchida, T., Takahashi, K., Ueda, H.: Discovery of Frequent Tag Tree Patterns in Semistructured Web Documents. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, pp. 341–355. Springer, Heidelberg (2002)
11. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In: Proceedings of the 17th International Conference on Data Engineering (ICDE), pp. 215–224 (2001)
12. Wang, J., Han, J., Pei, J.: CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In: Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 236–245 (2003)
13. SPMF: A Sequential Pattern Mining Framework, http://www.philippe-fournier-viger.com/spmf/
14. Kurt, A., Tozal, E.: Classification of XSLT-Generated Web Documents with Support Vector Machines. In: Nayak, R., Zaki, M.J. (eds.) KDXD 2006. LNCS, vol. 3915, pp. 33–42. Springer, Heidelberg (2006)
15. Sigmod Record in XML, http://www.sigmod.org/publications/sigmod-record/xml-edition
16. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley-Interscience (1990)
17. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34(1), 1–47 (2002)

Context-Aware Personalized Search Based on User and Resource Profiles in Folksonomies

Haoran Xie, Qing Li, and Xudong Mao

Department of Computer Science, City University of Hong Kong, Hong Kong
hrxie2@student.cityu.edu.hk, {itqli,xudonmao}@cityu.edu.hk

Abstract. The explosion of collaborative tagging data nowadays prompts an urgent demand upon Web 2.0 communities in assisting users to search interested resources quickly and effectively. Such a requirement entails much research on utilization of tag-based user and resource profiles so as to provide a personalized search in folksonomies. However, one major shortage for existing methods is their uniform treatment of user profile in the same way for each query, hence the search context for each query is ignored. In this paper, we focus on addressing this problem by modeling the search context. To capture and understand user intention, a nested context model is proposed. Furthermore, we conduct the experimental evaluation upon a real life data set, and the experimental result demonstrates that our approach is more effective than baselines.

1 Introduction

With the explosion of collaborative tagging data nowadays in various Web 2.0 communities such as web page bookmark collection (Del.icio.us[1]), movie recommendation (Movielens[2]) and image sharing (Flickr[3]), there is an urgent demand in assisting users to search interested resources quickly and effectively. Such a requirement entails much research on utilization of tag-based user and resource profiles so as to provide a personalized search in folksonomies [3,13,16,21]. However, these methods have a shortage of neglecting user query intention, and let us look at the following example as shown in Fig.1.

Example 1. *Suppose that user Alice annotated two different recipes by two tags "icecream" and "spicy" . If we adopt the tag-based profiling methods (suppose we use term frequency to construct user profile), the user profile of Alice can be constructed as $\overrightarrow{U}_{Alice} = (icecream : 1, spicy : 1)$. And there are three recipes i, j and k whose profiles are described below.*

$$\overrightarrow{R}_i = (spicy : 1, beef : 1, braised : 1)$$

$$\overrightarrow{R}_j = (beef : 1, lettuce : 1)$$

---

[1] http://www.delicious.com/
[2] http://www.movielens.org/
[3] http://www.flickr.com/

$$\overrightarrow{R}_k = (icecream : 1, sweet : 1)$$

When Alice tries to search some recipes and input keywords "beef" and "braise", the above three recipes will all be returned even though recipe k is not relevant to her query at all. This is because the resource profile of recipe k matches the tag "icecream" in the user profile of Alice. The problem is due to that "icecream" in Alice's profile is not relevant to her issued query. In other words, tags in her profile such as "icecream" should have different degrees of relevance to different queries context. In this particular example, in the search context of finding recipes about "braised beef", the tag "icecream" should not be counted at all.
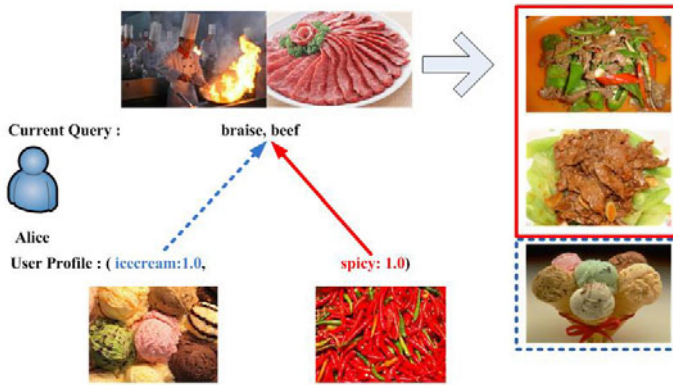
Fig. 1. An example of useless tag in user profile for current query

Therefore, we consider that it is unreasonable to use a static relevance for all tags in user profile for each issued queries. To address this problem, in this paper, we propose a search context model to capture and understand user intention. Since queries may have different search contexts, and they can be utilized to measure how relevant each tag in user profile to the current query is. The contributions of this paper include the following: (i) We propose an explicit nested context model in order to capture and understand user intention; (ii) By utilizing the nested context model, we devise a context-aware personalized search approach to adjust the relevance of tags in a user profile dynamically, so as to rank resources based on user preferences and the search contexts; (iii) Experiments are conducted on real-life data set by comparing the performance of our proposed approach and baseline methods. The experimental results are consistent with our observations.

The rest of this paper is organized as follows. In Section 2, we review some related works in collaborative tagging systems, context modeling and personalized search. In Section 3, our model for constructing user and resource profiles, and the nested search context modeling, are described. We present a context-aware personalized search approach to adjust the relevance of tags in user profile dynamically in Section 4. Experiments on a real life data set are conducted, and

the results are analyzed in Section 5. Finally, we summarize our work and discuss some future research directions in Section 6.

2   Related Works

In this section, we review some related works in collaborative tagging systems, context modeling and personalized search.

Collaborative Tagging Systems. Literature on collaborative tagging can be generally classified into two categories. One is about the investigation and analysis of the characteristics of user generated tags. In [6], Golder and Huberman studied tag usage patterns and user behavior in tagging.In an attempt to figure out valuable tags for search, Bischoff et al. [2] conducted a survey on some real tagging data sets. Manish et al. [7] also carried out a comprehensive survey on various features about social tagging data and techniques. The other is to discover features like link structure and semantic similarities in the folksonomies for various applications. The two novel algorithms proposed by Bao et al. [1], consisting of SocialSimRank (SSR) and SocialPageRank (SPR), incorporated benefits from social annotations so as to assist web search. [12] examined three (naive, co-occurrence and adaptive) approaches to construct the tag-based profile and their comparison.

Context in IR. In IR research community, context modeling has attracted much attention, and been utilized to understand user search intentions. Hassan et al. [8] theorized context that can be used in information retrieval. Besides, the context-sensitive search models derived from user search log were developed by Shen et al. [14,15]. In [18], the pre-query session activity was also regarded as the context in order to predict user short-term interests. Based on the context model from cognitive science by [11], White et al. [17] compared various contextual sources, and found that their hybrid was the most effective in terms of predicting user behavior. Sheng et al. [20] specified the context as four kinds of relationships: reformulation, specialization, generalization, and general association between previous queries and current one to facilitate Web search result ranking. Furthermore, the context can be adopted in some other applications such as query classification [5], query prediction [9] and Web page recommendation [17].

Personalization in Folksonomies. There are several relative research efforts on the utilization of resource profile and user profile for the facilitation of personalized search in folksonomies. The term frequency (TF) profiles proposed by Noll and Meinel [13] aims to discover related tags for users and resources, hence personalized ranking is provided. The term frequency-inverse document frequency (TF-IDF), Best Matching 25 (BM25) [21] and their hybrid [16] paradigms are followed by later works. Taken into account of the two kinds of sources, TF-IDF is combined with the user and resource profiles along with positions of tags in [4]. To further improve the previous tag-based methods, a normalized term frequency (NTF) was devised to model user and resource profiles in our earlier

work [3]. However, these methods utilize user profiles in a static way, so that the search contexts are ignored for different queries.

3   The Proposed Model

3.1   Tag-Based User and Resource Profiling

In a Web 2.0 collaborative tagging community, a user tag on a particular resource signifies two things. On the one hand, the tag can reflect this user's preference in some sense. On the other hand, the tag can be used to describe the resource's feature to certain extent. Based on these two observations and existing forms of tag-based profile in folksonomies, we define user and resource profiles as follows.

Definition 1. *A **user profile** of user a, represented by $\overrightarrow{U_a}$, is a vector in the form of tag:value pairs, i.e.,*

$$\overrightarrow{U_a} = (t_{a,1} : v_{a,1}, t_{a,2} : v_{a,2}, \cdots, t_{a,n} : v_{a,n})$$

where $t_{a,x}$ is a tag used to annotate some resources by user a, n is the total number of all tags used by the user, $v_{a,x}$ indicates the degree of preference for user a on tag $t_{a,x}$. According to our earlier work [3], NTF (Normalized Tag Frequency) paradigm is the most suitable in calculating the degree of preference[4], namely:

$$v_{a,x} = \frac{N_{a,x}}{N_a} \tag{1}$$

where $N_{a,x}$ is the frequency of the use of tag x to annotate resources by user a, and N_a is the amount of resources tagged by user i. The higher the value of $v_{a,x}$, the greater preference to tag x by user a.

Definition 2. *A **resource profile** of resource i, represented by $\overrightarrow{R_i}$, is a vector in the form of tag:value pairs:*

$$\overrightarrow{R_i} = (t_{i,1} : w_{i,1}, t_{i,2} : w_{i,2}, \cdots, t_{i,n} : w_{i,n})$$

where $t_{i,x}$ is a tag having been used by some users to describe resource i, n denotes the total number of tags annotating resource i, $w_{i,x}$ is a relevant degree of $t_{i,x}$ to resource i. Similarly, it can be calculated by NTF as follows:

$$w_{i,x} = \frac{M_{i,x}}{M_i} \tag{2}$$

where $M_{i,x}$ is the number of users using tag x to annotate resource i, and M_i is the total amount of users having annotated resource i. The higher the value of $w_{i,x}$, the more salient or relevant tag x for resource i.

---

[4] To confirm this conclusion, we will compare NTF to other paradigms such as TF-IDF, BM25 and their Hybrid in our experiments.

3.2   Nested Context Modeling

The purpose of the user profiling in personalized search is to predict what potential tags users may prefer. However, as discussed in **Example 1**, tags in a user profile should not be static to different queries issued by the user. To tackle this problem, we identify the context for the current issued query, and then use it to measure the relevance of each tag in user profile to current query. In this subsection, based on context model in [10,11], our proposed nested context model is introduced. As shown in Fig. 2, there are three different levels of contexts in the nested model:
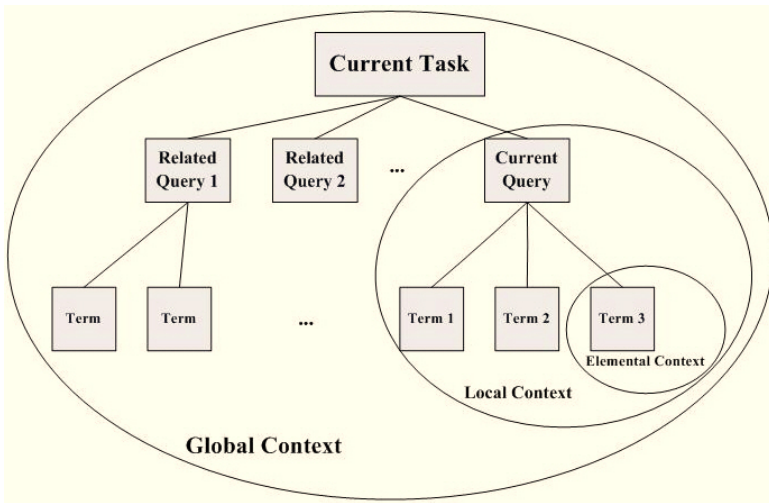
Fig. 2. The nested context modeling

Elemental Context (Term/Tag Level). Essentially, the elemental context for a query term/tag is a set of tags relevant to the query term. In a collaborative tagging system, if two different tags are used to annotate the same resource by some users, we assume that these two tags are relevant. According to this assumption, to obtain the elemental context for the query term, we consider that tags co-occurs with the query term in a resource profile should be relevant, and let us revisit **Example 1** to see what are elemental contexts for each query terms. As illustrated in Fig. 3, the elemental context for query term "beef" is the set of the co-occurred tags which include $\{spicy, lettuce, braise\}$. Similarly, the elemental context for query term "braised" is $\{spicy, beef\}$. We can see that the tag "icecream" in the user profile does not appear in these two elemental contexts. If we adopt these two elemental contexts to measure relevance for all tags, the tag "icecream" should be irrelevant so that recipe k will not be regarded as relevant resource to current query. Formally, the elemental context for a query term/tag is defined as follows.
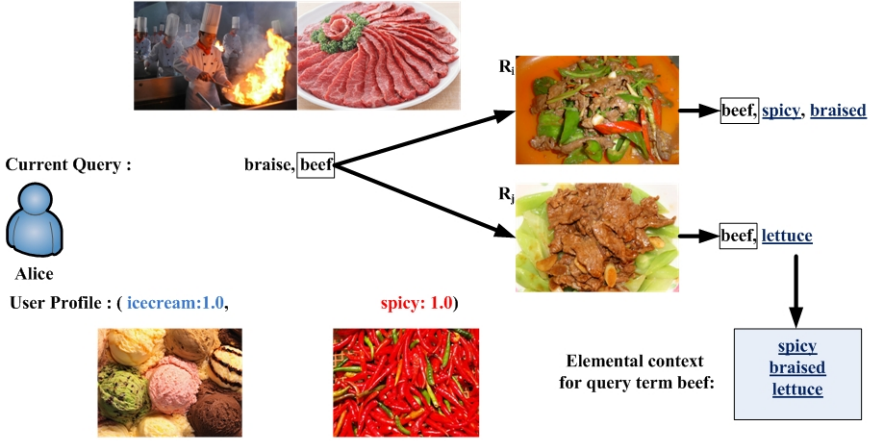
Fig. 3. An example of elemental context for a query term

Definition 3. *An **elemental context** for a query term/tag t_q^i, denoted by E_i, is a set of tags/terms which co-occur in the same resource:*

$$E_i = \{t_x | \exists x, \exists y, t_x \in R_y, t_q^i \in R_y\}$$

where t_q^i is a term of the current query, t_x is the tag co-occurred with t_q^i in any resource profile.

Local Context (Query Level). The local context is at the query level, and considers all terms/tags in a query. We define the local context for a query as follows.

Definition 4. *A **local context** for a query Q_j, denoted by L_j, is a union set of all elemental contexts for each term in the query:*

$$L_j = \{t_x | t_x \in \bigcup_{\forall i}^{t_q^i \in Q_j} E_i\}$$

where t_x is a tag in the elemental context for a particular query term, t_q^i is a query term in the query Q_j. For example, the local context for the query in **Example 1** is the union set $\{spicy, lettuce, braise, beef\}$ of elemental context for "beef" $\{spicy, lettuce, braise\}$ and "braised" $\{spicy, beef\}$.

Global Context (Task Level). The global context is at the task level, and a task is defined by all queries issued by a user within the current session. We demarcate the session by the 30-minute threshold as in other Web log analysis [19]. Accordingly, we define the global context for a task as follows.

Definition 5. *A **global context** for a task T, denoted by G_k, is a union set of all local contexts for each query in the task:*

$$G_k = \{t_x | t_x \in \bigcup_{\forall j}^{Q_j \in T} L_j\}$$

where T is the task for the current query, t_x is a tag occurred in any of the queries' local context. The global context is to identify how valuable a tag in a user profile to the current query is, thereby facilitating the personalized search.

4   Personalized Search

After the identification of the global context, the next major step is to achieve the personalized search for the query issuer. This process can be further divided into two sub-processes. The first one is to measure how relevant or valuable all tags in a user profile under the current global context is, so as to adjust the relevant degree of these tags for the query[5]. The second process is to rank resources by considering resource profiles, the contextualized user profile and the query.

4.1   Contextualization

Contextualization is the process of measuring how relevant (or valuable) a tag in a user profile to current context is. To achieve this goal, we need to measure the relevance between the tag and the elemental context firstly. Based on our observation in co-occurred tags in resource profiles, we make the following assumption.

Assumption 1. *For two tags x and y and query term q, if tag x has more frequent co-occurrence with query term q in the resource profiles than tag y, then tag x should have greater relevance than tag y to the elemental context of query term q.*

According to the above assumption, the relevance between a tag and the elemental context of a query term (**elemental context relevance** for short) is calculated as follows:

$$Rel(E_i, t_x) = \frac{|\{R_y | t_x \in R_y, t_q^i \in R_y\}|}{N} \tag{3}$$

where $|\{R_y | t_x \in R_y, t_i \in R_y\}|$ is the number of the set of resources which tag t_x and query term t_q^i co-occur, N is the total number of resources in the collaborative tagging system. Then, the relevance between a tag and the local context of a query (**local context relevance** for short) is obtained by the average of its all elemental context relevances, as follows:

$$Rel(L_j, t_x) = \frac{\sum_{i=1}^{N_{Q_j}} Rel(E_i, t_x)}{N_{Q_j}} \tag{4}$$

---

[5] We name this sub-process as "contextualization" and adjusted user profile as "contextualized user profile".

where N_{Q_j} is the number of query terms in query Q_j. Finally, the relevance between a tag and the global context of a task (**global context relevance** for short) is measured as follows:

$$Rel(G_k, t_x) = \frac{\sum_{j=1}^{N_{T_k}} Sim(Q_c, Q_j) \cdot Rel(L_j, t_x)}{N_{T_k}} \quad (5)$$

where N_{T_k} is the number of queries in the current task T_k, $Rel(L_j, t_x)$ is the local context relevance, $Sim(Q_c, Q_j)$ is the similarity between a query in the task and current query. Intuitively, the similarity can be calculated by the Jaccard Similarity between two queries as follows:

$$Sim(Q_c, Q_j) = \frac{|Q_c \cap Q_j|}{|Q_c \cup Q_j|} \quad (6)$$

However, this method has a shortage, as revealed by the following example.

Example 2. *Suppose that user Bob has issued two related queries $\overrightarrow{Q}_1 = \{sweet, food\}$ and $\overrightarrow{Q}_2 = \{icecream\}$, and there are two recipes i and j whose profiles are given below:*

$$\overrightarrow{R}_i = (sweet : 1, icecream : 1, cold : 1)$$

$$\overrightarrow{R}_i = (sweet : 1, cake : 1)$$

If we use Jaccard Similarity to measure the relevance between these two queries, it will be zero. This method is unreasonable since the intentions behind these two queries are quite similar. The Jaccard similarity ignores the case where two queries are related but have no overlapping query term.In contrast, the local context L_1 for Q_1 is $\{icecream, cold, cake\}$ and L_2 for Q_2 is $\{sweet, cold\}$, so if we further take the local context together with query terms into consideration, these two queries will be similar to a certain extent, which is more reasonable.

Based on the above analysis, we make the following assumption:

Assumption 2. *For any three queries Q_i, Q_j and Q_c, if the local context of Q_i is more relevant than Q_j to the local context of Q_c, then query Q_i should be more relevant (or similar) than Q_j to query Q_c.*

Based on the above assumption, we propose a **revised query similarity** measurement between two queries as follows.

$$Sim^*(Q_c, Q_j) = \frac{|\{Q_c \cup L_c\} \cap \{Q_j \cup L_j\}|}{|\{Q_c \cup L_c\} \cup \{Q_j \cup L_j\}|} \quad (7)$$

To revisit **Example 2**, we can calculate that similarity between two queries is 0.6 which are more reasonable than using Jaccard similarity. When the global context relevance is measured, we can adjust the existing weight values in the user profile, the result of which is named as contextualized user profile. Formally, we define a contextualized user profile as follows.

Definition 6. *A* **contextualized user profile** *under a query's global context G_k by user a, represented by $\overrightarrow{U_a^k}$, is a vector in the form of tag:value pairs,*

$$\overrightarrow{U_a^k} = (t_{a,1} : v_{a,1}^k, t_{a,2} : v_{a,2}^k, \cdots, t_{a,n} : v_{a,n}^k)$$

where $t_{a,x}$ is a tag in the user profile, $v_{a,x}^k$ is the contextualized weight value of tag x, and $v_{a,x}^k$ can be obtained as follows:

$$v_{a,x}^k = Rel(G_k, t_x) \cdot v_{a,x} \qquad (8)$$

where $v_{a,x}$ is the degree of preference for user a on tag $t_{a,x}$ according to **Definition 1**, $Rel(G_k, t_x)$ is the degree of relevance for tag t_x under global context G_k. A greater value of $v_{a,x}^k$ means greater relevance to current query and greater preference by the user.

4.2 Resource Ranking

In this section, we discuss how to rank resource based on resource profiles, the given contextualized user profile and query. This problem can be formally described by the following function:

$$\theta : U \times Q \times R \to [0, 1] \qquad (9)$$

where U is the set of users, Q is the set of queries, and R is the set of resources. The result of θ function is a relevant ranking score of a resource to a query issued by a user. The greater the score, the more relevant the resource to the query.

In some earlier work [3,4], this problem is divided into query relevance $\delta : Q \times R \to [0, 1]$ and user interest relevance $\gamma : U \times R \to [0, 1]$. Intuitively, these two relevances can adopt the cosine similarity as follows.

$$\theta(\overrightarrow{U_a^k}, \overrightarrow{Q}, \overrightarrow{R_i}) = \cos(\overrightarrow{U_a^k}, \overrightarrow{R_i}) \cdot \cos(\overrightarrow{Q}, \overrightarrow{R_i}) \qquad (10)$$

where *cos* function is the cosine similarity between two vectors. However, it has been proven less effective than the fuzzy function in our earlier work [3], namely:

$$\theta(\overrightarrow{U_a^k}, \overrightarrow{Q}, \overrightarrow{R_i}) = \frac{\frac{s}{m} \cdot \frac{\sum w_{i,x}}{m} + \frac{\sum l_x \cdot v_{a,x}^k}{m}}{2}, t_{i,x} \in \overrightarrow{Q} \qquad (11)$$

where s is the number of terms matching the query, m is the number of query terms, and

$$l_x = \begin{cases} w_{i,x} + (1 - v_{a,x}^k)(1 - w_{i,x}), & w_{i,x} \in (0, 1), v_{a,x}^k > 0 \\ 1, & w_{i,x} = 1, v_{a,x}^k > 0 \\ 0, & w_{i,x} = 0, v_{a,x}^k > 0 \end{cases} \qquad (12)$$

The personalized ranking is based on the relevance score of θ function. In other words, the greater value of relevance score, the higher ranking the resource in the search result list.

5   Experiment

5.1   Data Set and Metrics

Our data is collected from our implemented prototype: Folksonomy-based Multi-media Retrieval System (FMRS), which is specifically designed for personalized recipe search. This data set includes 500 recipes, 7889 tags and 203 users, each of whom tagged 16.7 recipes on average. The tags describe various aspects of the recipes and users' perception on them. To evaluate our proposed method, the data set is randomly split into two parts: 80% as training and 20% as test set. We use two metrics $P@N$ (Precision @N) [17] and MRR (Mean reciprocal rank). The $P@N$ indicates how accurate a particular personalized search strategy is, while MRR measures how fast this personalized strategy assists users to find the target resource.

5.2   Baselines

As shown in Table 1, we have six different combinations of strategies to be compared in the experiments. The Context Type refers to the three kinds of contexts as defined in **Definitions 3 to 5**. Moreover, there are two query similarity measurements proposed (**Equations 6 and 7**). Furthermore, for ranking methods, there are cosine and fuzzy methods (ref. **Equations 10 and 11**). In addition, for each strategy combination, four paradigms are adopted as baseline methods, namely, TF-IDF, BM25 [21], HYBRID[6] [16] and NTF [3] for constructing tag-based user and resource profiles. Therefore, as shown in Table 2, we have totally 24 methods for comparison.

Table 1. Abbreviation of different strategies

Abbreviation	Context Type	Query Simlarity Measurement	Ranking Method
N	No Context	NA	Cosine
E	Elemental Context	NA	Cosine
L	Local Context	NA	Cosine
G	Global Context	Jaccard Similarity	Cosine
$G^{\dagger}$	Global Context	Revised Query Similarity	Cosine
G^{*}	Global Context	Revised Query Similarity	Fuzzy

5.3   Experimental Results

The experimental results in terms of metrics $P@N$ and MRR for the 24 methods are shown in Table 2. Generally, we can observe that those methods with context information outperform conventional methods (all N methods) which consider no context information. It means that context user profiling is more precise and effective than conventional static way (**Example 1**). Indeed, the more complete

---

[6] The hybrid of TF-IDF and BM25.

Table 2. Performance Comparison

	N				E				L			
	BM25	TFI	HYB	NTF	BM25	TFI	HYB	NTF	BM25	TFI	HYB	NTF
P@5	0.092	0.105	0.105	0.192	0.090	0.105	0.106	0.194	0.092	0.105	0.109	0.197
P@10	0.113	0.113	0.170	0.366	0.115	0.114	0.170	0.363	0.116	0.115	0.173	0.368
P@20	0.133	0.127	0.279	0.449	0.134	0.133	0.281	0.453	0.133	0.134	0.282	0.450
MRR	0.094	0.112	0.118	0.198	0.096	0.114	0.119	0.207	0.096	0.115	0.122	0.215
	G				$G^{\dagger}$				**G^{*}**			
	BM25	TFI	HYB	NTF	BM25	TFI	HYB	NTF	BM25	TFI	HYB	**NTF**
P@5	0.096	0.109	0.109	0.203	0.111	0.118	0.120	0.207	0.120	0.129	0.133	**0.211**
P@10	0.117	0.117	0.178	0.374	0.122	0.124	0.186	0.376	0.125	0.133	0.189	**0.380**
P@20	0.134	0.142	0.290	0.456	0.147	0.150	0.299	0.462	0.153	0.161	0.314	**0.472**
MRR	0.100	0.118	0.131	0.225	0.125	0.129	0.141	0.236	0.131	0.142	0.157	**0.243**

context used by the method, the better performance gains ($G > L > E > N$[7]) by all the four profile constructing paradigms. In other words, the context at a high level has more complete information, so can achieve better performance than those at lower levels (e.g., Global versus Local and Elemental). This result verifies the rationale of our proposed nested context model. Furthermore, $G^{\dagger} >$ G signifies that **revised query similarity** is more reasonable than Jaccard similarity (**Example 2**). Besides, no matter what the strategy combination is used, the NTF always gain the best performance over all four paradigms. And $G^{*} >$ G indicates that the fuzzy resource ranking method is more effective than the cosine one. These two findings are consistent with the conclusions in our previous work [3].

6   Conclusion

In this paper, we focus on tackling the problem caused by neglecting user search context (treating the relevance degree of tags in user profile as static to each query) in conventional personalized methods in folksonomies. A nested context model is proposed to capture different levels of search contexts to solve this problem. Besides, we have discussed how to contextualizing a user profile under a context. By conducting the experiment, we have shown that the context information and our proposed nested context model can be quite effective and valuable to personalized search in a collaborative tagging system. There are some future directions for our upcoming research. One is for us to incorporate additional information such as user community, social relationships to explore more effective and powerful personalized search. Another possibility is to devise and try other user and resource profiling schemes such as multi-layered or graph-based ones.

---

[7] E > N means that E methods outperform N methods.

References

1. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: WWW 2007, pp. 501–510. ACM (2007)
2. Bischoff, K., Firan, C.S., Nejdl, W., Paiu, R.: Can all tags be used for search? In: CIKM 2008, pp. 193–202. ACM (2008)
3. Cai, Y., Li, Q.: Personalized search by tag-based user profile and resource profile in collaborative tagging systems. In: CIKM 2010, pp. 969–978. ACM (2010)
4. Cai, Y., Li, Q., Xie, H., Yu, L.: Personalized Resource Search by Tag-Based User Profile and Resource Profile. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 510–523. Springer, Heidelberg (2010)
5. Cao, H., Hu, D.H., Shen, D., Jiang, D., Sun, J.-T., Chen, E., Yang, Q.: Context-aware query classification. In: SIGIR 2009, pp. 3–10. ACM (2009)
6. Golder, S.A., Huberman, B.A.: Usage patterns of collaborative tagging systems. J. Inf. Sci. 32, 198–208 (2006)
7. Gupta, M., Li, R., Yin, Z., Han, J.: Survey on social tagging techniques. SIGKDD Explor. Newsl. 12, 58–72 (2010)
8. Hassan, A., Jones, R., Klinkner, K.L.: Beyond dcg: user behavior as a predictor of a successful search. In: WSDM 2010, pp. 221–230. ACM (2010)
9. He, Q., Jiang, D., Liao, Z., Hoi, S., Chang, K., Lim, E., Li, H.: Web query recommendation via sequential query prediction. In: ICDE 2009, pp. 1443–1454. IEEE (2009)
10. He, Q., Pei, J., Kifer, D., Mitra, P., Giles, L.: Context-aware citation recommendation. In: WWW 2010, pp. 421–430. ACM (2010)
11. Ingwersen, P., Järvelin, K.: The Turn: Integration of Information Seeking and Retrieval in Context. Springer-Verlag New York, Inc. (2005)
12. Michlmayr, E., Cayzer, S.: Learning user profiles from tagging data and leveraging them for personal (ized) information access. In: WWW 2007. Citeseer (2007)
13. Noll, M.G., Meinel, C.: Web Search Personalization Via Social Bookmarking and Tagging. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 367–380. Springer, Heidelberg (2007)
14. Shen, X., Tan, B., Zhai, C.: Context-sensitive information retrieval using implicit feedback. In: SIGIR 2005, pp. 43–50. ACM (2005)
15. Tan, B., Shen, X., Zhai, C.: Mining long-term search history to improve search accuracy. In: KDD 2006, pp. 718–723. ACM (2006)
16. Vallet, D., Cantador, I., Jose, J.M.: Personalizing Web Search with Folksonomy-Based User and Document Profiles. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 420–431. Springer, Heidelberg (2010)
17. White, R.W., Bailey, P., Chen, L.: Predicting user interests from contextual information. In: SIGIR 2009, pp. 363–370. ACM (2009)
18. White, R.W., Bennett, P.N., Dumais, S.T.: Predicting short-term interests using activity-based search context. In: CIKM 2010, pp. 1009–1018. ACM (2010)
19. White, R.W., Drucker, S.M.: Investigating behavioral variability in web search. In: WWW 2007, pp. 21–30. ACM (2007)
20. Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., Li, H.: Context-aware ranking in web search. In: SIGIR 2010, pp. 451–458. ACM (2010)
21. Xu, S., Bao, S., Fei, B., Su, Z., Yu, Y.: Exploring folksonomy for personalized search. In: SIGIR 2008, pp. 155–162. ACM (2008)

Distance-Based Outlier Detection on Uncertain Data of Gaussian Distribution

Salman Ahmed Shaikh and Hiroyuki Kitagawa

Graduate School of Systems and Information Engineering
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan
salman@kde.cs.tsukuba.ac.jp, kitagawa@cs.tsukuba.ac.jp
https://www.kde.cs.tsukuba.ac.jp/

Abstract. Managing and mining uncertain data is becoming important with the increase in the use of devices responsible for generating uncertain data, for example sensors, RFIDs, etc. In this paper, we extend the notion of distance-based outliers for uncertain data. To the best of our knowledge, this is the first work on distance-based outlier detection on uncertain data of Gaussian distribution. Since the distance function for Gaussian distributed objects is very costly to compute, we propose a cell-based approach to accelerate the computation. Experimental evaluations of both synthetic and real data demonstrate effectiveness of our proposed approach.

Keywords: Outlier Detection, Uncertain Data, Cell Based Approach.

1 Introduction

Outlier detection is one of the most important data mining techniques with a vital importance in many application domains including credit card fraud detection, network intrusion detection, environmental monitoring, medical sciences, etc. Although there exists no any universally agreed upon definition of outliers, some definitions are general enough to give a basic idea of outliers. Hawkins [5] defines an outlier as an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. In [6] Barnet and Lewis mentioned that an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.

Most of the earliest outlier detection techniques were given by statistics. In statistics over 100 outlier detection techniques have been developed for different circumstances, depending on the data distribution, whether or not the distribution parameters are known, the number of expected outliers and the type of expected outliers [6], but most statistical techniques are univariate and in majority of techniques, the parameter of distribution may be difficult to determine. In order to overcome problems in statistical techniques several distance-based approaches for outlier detection are proposed in computer science [3], [7], [8].

Uncertainty. Due to the incremental usage of sensors, RFIDs and similar devices for data collection these days, data contains certain degree of inherent uncertainty. The causes of uncertainty may include limitation of equipments, absence of data and delay or loss of data in transfer. In order to get reliable results from such a data, uncertainty needs to be considered in calculation. In this work we propose a notion of distance-based outliers on uncertain data.

In our work, uncertainty of data is modelled by the most commonly used PDF i.e., Gaussian distribution. We have derived a distance function using Gaussian difference distribution to compute the distance-based outliers on uncertain data. Our distance function includes the integral of irreducible function, which makes the distance function computation very costly. Therefore we also propose a cell-based algorithm of outlier detection to efficiently compute the distance-based outliers on uncertain data. The cell-based algorithm prunes objects by identifying outliers or pruning non-outliers without the need to compute costly distance function, hence reducing the number of distance function evaluation required. Finally we make use of grid structure to further reduce the computation time required for distance-based outlier detection.

The rest of the paper is organized as follows. In section 2, the related work is discussed. Section 3 gives the derivation of distance function and the naive algorithm of distance-based outlier detection on uncertain data. The cell-based algorithm is given in section 5. Section 6 is dedicated to empirical study and Section 7 concludes our paper.

2   Related Work

Outlier detection is a well studied area of data mining. Different authors have classified this area differently. The problem of outlier detection has been classified into statistical approaches, depth-based approaches, deviation-based approaches, distance-based approaches, density-based approaches and high-dimensional approaches by [9].

Distance-based outliers detection approach on deterministic data was introduced by Knorr et.al. in [3]. In this work, the authors defined a point p to be an outlier if at most M points are within d distance of the point. They also presented a cell-based algorithm to efficiently compute the distance-based outliers. [10] formulated distance-based outliers based on the distance of a point from its kth nearest neighbour. The points were ranked on the basis of its distance to its kth nearest neighbour and the top n points were declared outliers in this ranking. Recently in [7], the authors assessed and evaluated several distance-based outlier detection approaches and highlighted a family of state of the art distance-based outlier detection algorithms.

Recently a lot of research has focused on managing, querying and mining of uncertain data [11], [8], due to the use of sensors in many applications. The problem of outlier detection on uncertain data was first studied by Aggarwal et.al. in [11]. They represented an uncertain object by a PDF. They defined an uncertain object O to be a density-based (δ, η) outlier, if the probability of O

existing in some subspace of a region with density at least η is less than δ. In [8], the authors proposed the distance-based outlier detection on uncertain data. In their approach, each tuple in the uncertain table is associated with an existential probability. Moreover in their work, possible world semantic was used to mine the outliers. In our work, objects' uncertainty is modelled by Gaussian distribution and we utilize Gaussian difference distribution to calculate the outlier probability.

3   Distance-Based Outlier on Uncertain Data

Several definitions of distance-based outliers have been proposed in past. In this paper, we extend the notion of distance-based outliers given by E.M.Knorr et.al. in [3] for uncertain data of Gaussian distribution.

In statistics, the Gaussian distribution *(or normal distribution)* is the most important and the most commonly used distribution. In the following, we consider k-dimensional uncertain objects O_i, each given by a Gaussian PDF with attribute $\overrightarrow{\mathcal{A}_i} = (x_{i,1}, ..., x_{i,k})^T$, mean $\overrightarrow{\mu_i} = (\mu_{i,1}, ..., \mu_{i,k})^T$ and variance $\Sigma_i = diag(\sigma_{i,1}^2, ..., \sigma_{i,k}^2)$ respectively. The complete database consists of a set of such objects, $\mathcal{G}DB = \{O_1, ..., O_N\}$ where $N = |\mathcal{G}DB|$ is the number of uncertain objects in $\mathcal{G}DB$. The vector $\overrightarrow{\mathcal{A}_i}$ is a random variable of the corresponding uncertain objects that follows Gaussian distribution $\overrightarrow{\mathcal{A}_i} \sim \mathcal{N}(\overrightarrow{\mu_i}, \Sigma_i)$.

We assume that the observed coordinates are $\overrightarrow{\mu_i}$ vectors of the objects which follow Gaussian distribution. Based on this assumption, in the rest of the paper we will use $\overrightarrow{\mu_i}$ to denote the real observed coordinates of object O_i. We can now define the distance based outliers on uncertain data of Gaussian distribution as follows.

Definition. *An uncertain object O in a database $\mathcal{G}DB$ is a distance-based outlier, if the expected number of objects $O_i \in \mathcal{G}DB$ (including O itself) lying within d-distance of O is less than or equal to threshold $\theta = N(1 - p)$, where N is the number of uncertain objects in database $\mathcal{G}DB$, uncertain objects in $\mathcal{G}DB$ follow Gaussian distribution and p is the fraction of objects in $\mathcal{G}DB$ that lies farther than d-distance of O.*

According to the definition above, the set of uncertain distance-based outliers in $\mathcal{G}DB$ is defined as follows,

$$\mathcal{UDB}Outliers = \{O_i \in \mathcal{G}DB| \sum_{j=1}^{|\mathcal{G}DB|} Pr(|\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}| \le d) \le \theta\} \ . \qquad (1)$$

In order to find distance-based outliers in $\mathcal{G}DB$, the distance between Gaussian distributed objects need to be calculated. In the following we define and derive the expressions for difference between Gaussian distributed objects.

3.1   Gaussian Difference Distribution

The distribution of the difference of two Gaussian distributed variates O_i and O_j with means and variances (μ_i, σ_i^2) and (μ_j, σ_j^2) respectively, is given by another Gaussian distribution with mean $\mu_{i-j} = \mu_i - \mu_j$ and variance $\sigma_{i-j}^2 = \sigma_i^2 + \sigma_j^2$ [1]. Hence we can write $\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j} \sim \mathcal{N}(\mu_{i-j}, \sigma_{i-j}^2)$.

1-Dimensional Gaussian Difference Distribution within Distance d

Using Gaussian difference distribution, the probability that the uncertain object O_i lies within d-distance of uncertain object O_j is given by,

$$Pr(|\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}| \leq d) = \int\limits_{-d}^{d} \mathcal{N}(\mu_{i-j}, \sigma_{i-j}^2)\mathrm{d}x \ , \tag{2}$$

where $\overrightarrow{\mathcal{A}_i} \sim \mathcal{N}(\mu_i, \sigma_i^2)$ and $\overrightarrow{\mathcal{A}_j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$.

2-Dimensional Gaussian Difference Distribution within Distance d

The expression for the 2-dimensional Gaussian difference distribution is defined in Lemma 1 below.

Lemma 1.(2D Gaussian Difference Distribution within Distance d) *let* $\overrightarrow{\mathcal{A}_i} \sim \mathcal{N}(\overrightarrow{\mu_i}, \Sigma_i)$ *and* $\overrightarrow{\mathcal{A}_j} \sim \mathcal{N}(\overrightarrow{\mu_j}, \Sigma_j)$ *be two 2-dimensional Gaussian distributed variates, where* $\overrightarrow{\mu_i} = (\mu_{i,1}, \mu_{i,2})^T$, $\overrightarrow{\mu_j} = (\mu_{j,1}, \mu_{j,2})^T$, $\Sigma_i = diag(\sigma_{i,1}^2, \sigma_{i,2}^2)$ *and* $\Sigma_j = diag(\sigma_{j,1}^2, \sigma_{j,2}^2)$. *The probability that* O_i *lies within d-distance of* O_j *is given by,*

$$Pr(|\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}| \leq d) = \frac{1}{2\pi\sqrt{(\sigma_{i,1}^2 + \sigma_{j,1}^2)(\sigma_{i,2}^2 + \sigma_{j,2}^2)}}$$
$$\int_0^d \int_0^{2\pi} \exp\left\{-\left(\frac{(r\cos\theta - \alpha_1)^2}{2(\sigma_{i,1}^2 + \sigma_{j,1}^2)} + \frac{(r\sin\theta - \alpha_2)^2}{2(\sigma_{i,2}^2 + \sigma_{j,2}^2)}\right)\right\} r \, d\theta \, dr \ , \tag{3}$$

where $\alpha_1 = \mu_{i,1} - \mu_{j,1}$ *and* $\alpha_2 = \mu_{i,2} - \mu_{j,2}$ *are the differences between the means of objects* O_i *and* O_j.

Proof. See appendix.

Multidimensional Gaussian Difference Distribution within Distance d

Our distance function can easily be extended to multi-dimension case. Let $\overrightarrow{\mathcal{A}_i}$ and $\overrightarrow{\mathcal{A}_j}$ be two k-dimensional normal random vectors with means $\overrightarrow{\mu_i} = (\mu_{i,1}, ..., \mu_{i,k})^T$ and $\overrightarrow{\mu_j} = (\mu_{j,1}, ..., \mu_{j,k})^T$ and diagonal covariance matrices $\Sigma_i = diag(\sigma_{i,1}^2, ..., \sigma_{i,k}^2)$

and $\Sigma_j = diag(\sigma_{j,1}^2,...,\sigma_{j,k}^2)$ respectively. The probability that the uncertain object O_i lies within d distance of uncertain object O_j is given by,

$$Pr(|\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}| \le d) = \int_R \mathcal{N}(\overrightarrow{\mu_{i-j}}, \Sigma_{i-j})\mathrm{d}x , \tag{4}$$

where $\overrightarrow{\mu_{i-j}} = \overrightarrow{\mu_i} - \overrightarrow{\mu_j}$, $\Sigma_{i-j} = \Sigma_i + \Sigma_j$ and R is a sphere with centre $\overrightarrow{\mu_{i-j}}$ and radius d.

4   Naive Approach

The Naive approach of distance-based outlier detection on uncertain data is the use of Nested-loop. The approach includes the evaluation of distance function between each object $O_i \in \mathcal{G}DB$ and every other object in the $\mathcal{G}DB$ until O_i may be decided as outlier or non-outlier. In the worst case this approach requires the evaluation of $O(N^2)$ distance functions. The algorithm 1 gives the naive approach of distance based outliers.

Algorithm 1. Distance-based Outlier on Uncertain Data: The NL Approach

---

Input: database $\mathcal{G}DB$, distance d, percentage p , standard deviation σ
Output: Uncertain Distance Based Outliers
1: $N \leftarrow$ number of objects in $\mathcal{G}DB$;
2: $\theta \leftarrow N(1-p)$;   /*calculating the threshold value*/
3: **for each** O in $\mathcal{G}DB$ **do**
4:     $EV_O \leftarrow 0$;   /*EV_O denotes the expected value of object O*/

5:     **for each** O_i in $\mathcal{G}DB$ **do**
6:        $EV_O = EV_O + Pr(|\overrightarrow{\mathcal{A}} - \overrightarrow{\mathcal{A}_i}| \le d)$;
7:        **if** $EV_O > \theta$ **then**
8:           mark O as non-outlier, GOTO next O;
9:        **end if**
10:    **end for**
11:    mark O as outlier;
12: **end for**

---

5   Cell-Based Approach

The naive approach of distance-based outlier detection on uncertain data requires a lot of computational time to detect outliers even from small dataset. In the following we propose a cell-based approach of distance-based outlier detection on uncertain data, which can reduce significantly the number of distance functions evaluations. The proposed approach first map database objects to a cell-grid structure and then prunes majority of objects by identifying the cells containing only outliers or non-outliers. For un-pruned objects, Grid File indexing is utilized to further reduce the number of distance function computations.

5.1   Grid Structure

We assume that our data objects are 2 dimensional. In order to find distance-based outliers on uncertain data, we quantize each object $O_i \in \mathcal{GDB}$, in 2 dimensional space that has been partitioned into cells of length l (cell length is discussed in section 5.6). Let $C_{x,y}$ be any cell of the Grid, then the neighbouring cells of $C_{x,y}$ form layers around it as shown in Fig. 1. Layers of any cell $C_{x,y}$ in the Grid are defined as follows.

- Layer 1 cells of $C_{x,y}$ are given by

$$L_1(C_{x,y}) = \{C_{u,v} | u = x \pm 1, v = y \pm 1, C_{u,v} \neq C_{x,y}\} \, .$$

- Layer 2 cells of $C_{x,y}$, and are given by

$$L_2(C_{x,y}) = \{C_{u,v} | u = x \pm 2, v = y \pm 2,$$
$$C_{u,v} \notin L_1(C_{x,y}), C_{u,v} \neq C_{x,y}\} \, .$$

$L_3(C_{x,y}), ..., L_n(C_{x,y})$ are defined in a similar way. Where n denotes the number of cell layers and is discussed in section 5.6.
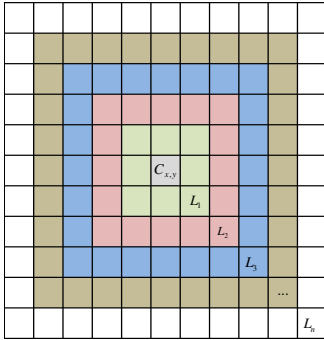
Fig. 1. Cell Layers
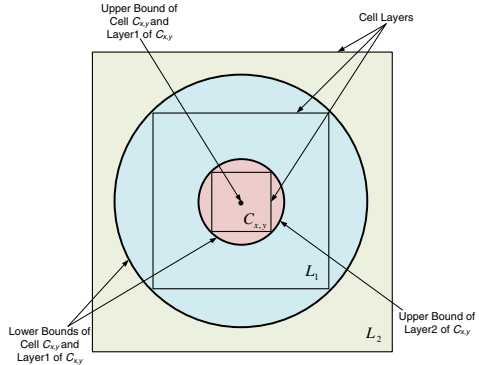
Fig. 2. Cell Bounds

5.2   Cell Layers Bounds and Lookup Table

The bounds of a cell or cell layers are defined for pruning outliers and non-outliers without evaluating the distance functions for the objects in the cell. The upper and lower bounds of any cell or cell layers are shown in Fig. 2 and are defined as follows.

Upper Bound. By an upper bound of a cell (cell layers) we mean the maximum contribution by any of the objects in this cell (cell layers) to the target cell. According to our distance function an object in cell $C_{x,y}$ can contribute at its maximum to object O in cell $C_{x,y}$ when $\alpha_x = \alpha_y = 0$ in Eq.3. Similarly the upper

bound contributions of objects in $L_i(C_{x,y})$ layers (i.e., $L_1(C_{x,y})$, ..., $L_n(C_{x,y})$) to objects in $C_{x,y}$ are obtained by setting $\alpha_x = \alpha_y = (i-1)\sqrt{2}l$ in Eq.3.

Lower Bound. By a lower bound of a cell (cell layers) we mean the minimum contribution by any of the objects in this cell (cell layers) to the target cell. According to our distance function an object in cell $C_{x,y}$ contributes at its minimum to object O in cell $C_{x,y}$ when $\alpha_x = \alpha_y = \sqrt{2}l$ in Eq.3. Similarly, the lower bound contributions of objects in $L_i(C_{x,y})$ layers (i.e., $L_1(C_{x,y})$, ..., $L_n(C_{x,y})$) to objects in $C_{x,y}$ are obtained by setting $\alpha_x = \alpha_y = (i+1)\sqrt{2}l$ in Eq.3.

Lookup Table. The above upper bound and lower bound of contributions of objects in $L_i(C_{x,y})$ to $C_{x,y}$ are decided only by the i-value and independent from the locations of $C_{x,y}$. Hence, we compute the bounds and store them in a lookup table to be used in the cell-based algorithm.

5.3 Pruning of Outliers and Non-outliers Cells

Having defined cell bounds and cell layers bounds, a cell can be pruned as an outlier or non-outlier cell. If the minimum contribution to cell $C_{x,y}$, obtained by the product of cell objects count and cell lower bound is greater than threshold θ, then none of the objects in $C_{x,y}$ could be outliers and we can prune it as non-outliers cell.

$$MinContribution(C_{x,y}) = 1 + (Count(C_{x,y}) - 1) * LowerBound(C_{x,y}) .$$

On the other hand if the maximum contribution to cell $C_{x,y}$, obtained by the product of cell objects count and cell upper bound plus the expected contribution by rest of the objects in the database $\mathcal{G}DB$ is less than or equal to θ, then all the objects in $C_{x,y}$ are outliers and we can prune it as outliers cell.

$$MaxContribution(C_{x,y}) = Count(C_{x,y}) + \\ (N - Count(C_{x,y})) * UpperBound(L_1(C_{x,y})) .$$

If none of the above conditions hold, then we need to check the contribution of higher cell layers i.e., contributions of $L_1(C_{x,y})$,...,$L_n(C_{x,y})$, until we may either decide the cell $C_{x,y}$ as containing only outliers or only non-outliers or left the cell undecided for the post-pruning evaluation.

5.4 Grid File Index

Cell-based pruning may leave some of the cells undecided, i.e., they are neither pruned as non-outliers cells nor as outliers cells. For all the uncertain objects in such cells, we need to follow Nested-loop approach. Our distance function of outlier detection requires a lot of computation time and may reduce the efficiency of our cell-based algorithm even after initial pruning. As we know from our distance function, that it produces higher probability for the nearer objects than the farther objects. We can utilize our Grid structure as Grid-file index [2] with

no additional indexing cost to retrieve the nearer objects before the farther objects for the computation of expected value of all un-pruned objects. This will further reduce the number of evaluations required for distance function, hence reducing the overall cost of computation.

5.5   Cell-Based Algorithm of Outlier Detection

In order to reduce the costly computation of distance function, we propose cell-based algorithm. The main idea of this algorithm is to prune the cells containing only outliers or non-outliers. Algorithm 2 starts by first calculating the bounds of cell layers and storing them in a look-up table. The database objects are then mapped to appropriate cells of the Grid. For each cell, $C_{x,y}$ in Grid, $MinContribution$ and $MaxContribution$ i.e., minimum and maximum contributions are maintained which are used for effectively pruning the cells as outliers or non-outliers. If a cell $C_{x,y}$ can not be pruned, the objects of such cells are checked individually for outliers using Grid-file index.

Although the number of distance function evaluations required in worst case for the cell-based algorithm is same as that of naive approach, i.e., $O(N^2)$ but the experimental results on both synthetic and real datasets show that our proposed approach is very efficient.

5.6   Cell Length l and Cell Layers n

Due to the complexity of our distance function, it is not possible to derive a single cell length l suitable for all the combinations of d and variances. Therefore we conducted several experiments to come up with a cell length which may produce efficient results.

A general observation from several experiments is that smaller the cell-length, shorter the execution time. Since smaller cell-length results in higher values cell bounds, which helps in pruning majority of objects during cell-based pruning stage and either very few or no cell is left for post-pruning evaluation, reducing the number of distance function evaluations. However very small cell length may also increase the execution time for cell-based algorithm as too small cell length results in a large number of cells and the time required to compute cell layers bounds increases. We need to check a few cell lengths before reaching the appropriate cell-length. A good starting point that we have found through experiments is $l = \frac{\sigma_1 + ... + \sigma_k}{k}$.

Cell Layers n. Since a Gaussian function decays exponentially with respect to the distance to its mean, the density contribution is small if the mean is far away from the target object. Using this fact, we conducted experiments where n was set to, 1) all layers in the Grid, 2) layers within $d + 6\sigma$ distance of target cells and 3) layers within $d + 3\sigma$ distance of target cells. We found that all three experiments retrieved same number of outliers and as expected, n with layers within $d + 3\sigma$ was faster than the other two. The reason for the same number of outliers in all three choices of n is that the contribution of the cells which are farther than $d + 3\sigma$ is negligibly small and has no effect on outlier detection.

Algorithm 2. Distance-based Outlier on Uncertain Data: Cell Based Approach

---

Input: database $\mathcal{G}DB$, distance d, percentage p , standard deviation σ
Output: Distance Based Outliers on Uncertain Data of Gaussian Distribution
 1: Compute and store cell bounds into lookup table using cell length l and maximum distance between any two objects in $\mathcal{G}DB$;
    /*Initialize the count $Count_i$ of each cell C_i in grid $Grid$*/
 2: **for each** C_i in $Grid$ **do**
 3:    $Count_i \leftarrow 0$;
 4: **end for**
    /*Mapping database objects to appropriate cells*/
 5: **for each** O in $\mathcal{G}DB$ **do**
 6:    map O to an appropriate cell C_i;
 7:    $Count_i \leftarrow Count_i+1$;   /*increase cell count by 1*/
 8: **end for**
 9: $\theta \leftarrow N(1-p)$;   /*calculating the threshold value*/

10: $n = \lceil \frac{max(|\overrightarrow{\mathcal{A}_p} - \overrightarrow{\mathcal{A}_q}|)}{l} \rceil$, where $O_p, O_q \in \mathcal{G}DB$;/*$n$ denotes the number of cell layers*/
    /*Pruning of outlier and non-outlier cells using cell layers' bounds*/
11: **for each** C_i in $Grid$ **do**
12:    **for** $j = 0 \to n$ **do**
13:        Calculate minimum and maximum contribution of cell C_i using upper and lower bounds respectively of 0 to j^{th} neighbouring cell layers of C_i;
14:        **if** $MinContribution(C_i) > \theta$ **then**
15:            prune C_i as non-outlier cell, GOTO Next C_i;
16:        **else if** $MaxContribution(C_i)+$ expected contribution of C_i from rest of the cell layers in $Grid \leq \theta$ **then**
17:            prune C_i as outlier cell, GOTO Next C_i;
18:        **end if**
19:    **end for**
20: **end for**
    /*Nested-loop approach using Grid File Index for objects in un-pruned cells*/
21: **for each** C_i in $Grid$ **do**
22:    **if** C_i not pruned as outlier or non-outlier cell and $Count_i \mathrel{!}= 0$ **then**
23:        **for each** O in C_i **do**
24:            $EV_O \leftarrow 0$;   /*EV_O denotes the expected value of object O*/

25:            **for each** O_j in C_i and higher layers of C_i in $Grid$ **do**
26:                $EV_O \leftarrow EV_O + Pr(|\overrightarrow{\mathcal{A}} - \overrightarrow{\mathcal{A}_j}| \leq d)$;
27:                **if** $EV_O > \theta$ **then**
28:                    O can not be outlier, GOTO next O;
29:                **end if**
30:            **end for**
31:            mark O as outlier;
32:        **end for**
33:    **end if**
34: **end for**

---

6   Empirical Study

We conducted extensive experiments on synthetic and real datasets to evaluate the effectiveness and accuracy of our proposed cell-based algorithm. All algorithms were implemented in C#, Microsoft Visual Studio 2008. All experiments were performed on a system with an Intel Core 2 Duo E8600 3.33GHz CPU and 2GB main memory running Windows 7 Professional OS. All programs run in main memory and no I/O cost is considered.

(a) NL vs. Cell-based   (b) Varying Cell Lengths   (c) Varying n for $l = 10$

(d) Varying n for $l = 30$ (e) Varying p (f) Varying d
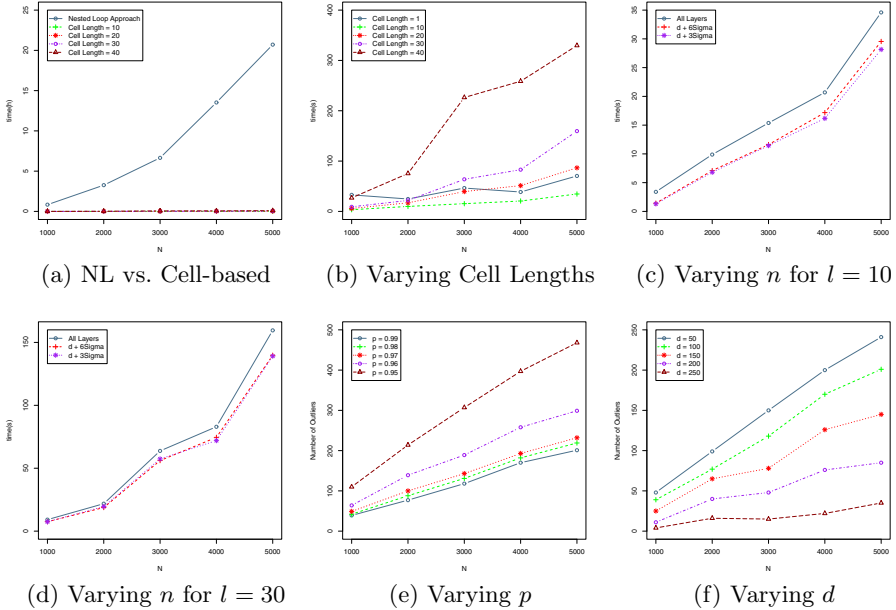
Fig. 3. Experiments on synthetic data (default $d = 100$, $p = 0.99$, $n =$ layers within $d + 3\sigma$ and $l = 10$)

6.1   Experiments on Synthetic Data

Unless specified, the experiments are performed on 5 uniformly distributed 2-dimensional datasets of sizes varying from 1000 to 5000 tuples respectively with parameters, distance d=100, p=0.99, $n =$ "layers within $d+3\sigma$ distance of target cell" and l=10. Uncertainty is simulated by representing each object as Gaussian distributed with means between 0 and 1000 and standard deviation $\sigma_x = 15$ and $\sigma_y = 15$ in x and y dimensions respectively.

It is obvious from Fig.3a that the time taken by the Nested-loop algorithm is very high and the execution time increases dramatically as the number of tuples in database increases.

Cell Length l and Cell Layers n. As discussed in section 5.6, smaller the cell length, shorter the execution time, which is obvious from Fig. 3b. However,

very small cell length may increase the execution time, due to the increase in time required for the computation of look-up table as shown in Fig. 3b for cell length = 1. From Fig.3c and Fig.3d, we can observe that cell layers within $d+3\sigma$ distance of target cell produces better results than all the layers in the Grid.

Varying Parameters p and d. Varying parameters p and d has an effect on number of outliers mined by the algorithm as shown by the plots in Fig.3e and Fig.3f respectively. Increasing p results in smaller threshold value, resulting in only a few and relatively stronger outliers.

Varying d has an effect on the distance function probability contribution. Larger d means larger number of objects may fall within d distance of object under consideration, hence increasing the expected value and reducing the number of outliers.

6.2   Experiments on Real Dataset

For experiments on real dataset, we used NBA Playoffs Player statistics from 1996 to 2006 available at [4]. The dataset contains the annual performance statistics of NBA players. The filtered dataset used in the experiments contain 2081 tuples, with threshold related parameter p set to 0.99. Therefore value of threshold $\theta = N(1-p) = 20.81$. Each player is represented as an uncertain object with 2 important statistics on his performance i.e., number of points and number of total rebounds. Both statistics are defined by means as the real observed values and standard deviations $\sigma_{points} = 20$ and $\sigma_{rebounds} = 10$.

Player Name	Team	year	Points Scored	Total Rebounds	Expected Value	Player Name	Team	year	Points Scored	Total Rebounds	Expected Value
Shaquille O'neal	LAL	1999	707	355	1.410357614	Dale Davis	IND	1999	190	263	8.737494804
Tim Duncan	SAS	2002	593	369	2.132579742	Shaquille O'neal	LAL	2003	473	291	9.200527361
Allen Iverson	PHI	2000	723	104	2.842902664	Dikembe Mutombo	PHI	2000	319	316	9.316349528
Michael Jordan	CHI	1997	680	160	3.538590847	Shaquille O'neal	LAL	2001	541	239	9.316349528
Dirk Nowitzki	DAL	2005	620	268	3.817675722	Karl Malone	UTA	1996	519	228	12.06860839
Ben Wallace	DET	2003	236	328	4.014292632	Karl Malone	UTA	1997	526	217	12.35327527
Dwyane Wade	MIA	2005	645	135	4.04669976	Kevin Garnett	MIN	2003	438	263	13.01360271
Ben Wallace	DET	2002	151	277	5.237158519	Reggie Miller	IND	1999	527	53	13.23169019
Ben Wallace	DET	2004	249	281	6.985909188	Shaquille O'neal	LAL	2000	487	247	13.23169019
Tim Duncan	SAS	2004	542	286	7.742466357	Kobe Bryant	LAL	2003	539	104	13.33036715
Dennis Rodman	CHI	1997	102	248	7.795215863	Kobe Bryant	LAL	2001	506	111	18.6784144
Michael Jordan	CHI	1996	590	150	8.051622656	Tim Duncan	SAS	2006	444	229	19.39066163

Fig. 4. NBA Players with Expected Value less than θ

Our experiments on NBA dataset mined the outstanding players during 1996 and 2006. From the expected values in Fig.4, Shaquille O'neal is the most outstanding player with maximum points scored and second maximum total rebounds. He has the outstanding performance from 1999 to 2003 except the year 2002. On the other hand, Tim Duncan's performance seems to decline during the course of his career as he was a strong outlier in 1999 and became weak outlier in 2006.

7   Conclusion and Future Work

In this paper, we extend the notion of distance-based outlier detection on uncertain data of Gaussian distribution. This is the first approach of distance-based outlier detection where the objects are modelled by Gaussian distribution. We derive distance function for distance-based outlier detection on uncertain data of Gaussian distribution and propose a cell-based algorithm to efficiently detect outliers by pruning majority of outliers and non-outliers cells. We also utilize grid-file index to further reduce the computation time required for the cell-based algorithm. Extensive experiments on synthetic and real data demonstrate the efficiency and scalability of our proposed algorithm.

In future, we are planning to extend this work in two dimensions. First, designing an adaptive algorithm with respect to cell length, in order to increase the efficiency of our cell-based algorithm. Second, expanding this work for uncertain data streams of Gaussian distribution.

References

1. Weisstein, E.W.: Normal Difference Distribution. From MathWorld - A Wolfram Web Resource,
   `http://www.mathworld.wolfram.com/NormalDifferenceDistribution.html`
2. Nievergelt, J., Hinterberger, H., Sevick, K.C.: The Grid File: An Adaptable, Symmetric multikey File Structure. ACM Transaction on Database Systems (1984)
3. Knorr, E.M., Ng, R.T., Tucakov, V.: Distance-Based Outliers: Algorithms and Applications. The VLDB Journal 8, 237–253 (2000)
4. NBA All-time Player Stats, Opendata by Socrata, `http://opendata.socrata.com`
5. Hawkins, D.: Identification of Outliers. Chapman and Hall (1980)
6. Barnett, V., Lewis, T.: Outliers in Statistical Data. John Wiley (1994)
7. Orair, G.H., Teixeira, C.H.C., Meira, W.: Distance-Based Outlier Detection: Consolidation and Renewed Bearing. Proc. of the VLDB Endowment (2010)
8. Wang, B., Xiao, G., Yu, H., Yang, X.: Distance-Based Outlier Detection on Uncertain Data. In: IEEE 9th International Conference on Computer and Information Technology (2009)
9. Kriegel, H.-P., Kröger, P., Zimek, A.: Outlier Detection Techniques. Tutorial at 16th ACM SIGKDD Conference (2010)
10. Ramaswamy, S., Rastogi, R., Shim, K.: Efficient Algorithms for Mining Outliers from Large Data Sets. In: Proceedings International Conference on Management of Data. ACM, SIGMOD (2000)
11. Aggarwal, C.C., Yu, P.S.: Outlier Detection with Uncertain Data. In: SIAM International Conference on Data Mining (2008)

Appendix

Proof of Lemma. Let O be a k-dimensional uncertain object with attributes $\overrightarrow{\mathcal{A}} = (x_1, ..., x_k)$, mean $\overrightarrow{\mu} = (\mu_1, ..., \mu_k)^T$ and a diagonal covariance matrix $\Sigma = diag(\sigma_1^2, ..., \sigma_k^2)$. The probability density function of O can be expressed as

$$f(\overrightarrow{\mathcal{A}}) = \frac{1}{\sqrt{(2\pi)^k det\Sigma}} exp\left\{ -\frac{(\overrightarrow{\mathcal{A}} - \overrightarrow{\mu})^T \Sigma^{-1} (\overrightarrow{\mathcal{A}} - \overrightarrow{\mu})}{2} \right\}.$$

Since Σ is diagonal, the distribution functions are independent in coordinates. Hence the k-dimensional normal distribution function is given by the product of k 1-dimensional normal distribution functions.

$$f(\overrightarrow{\mathcal{A}}) = \prod_{1 \leq i \leq k} \frac{1}{\sqrt{2\pi\sigma_i^2}} exp\left\{ -\frac{(x_i - \mu_i)^2}{2\sigma_i^2} \right\}.$$

Let O_i and O_j are two 2-dimensional uncertain objects with attributes $\overrightarrow{\mathcal{A}_i} = (x_{i,1}, x_{i,2})$ and $\overrightarrow{\mathcal{A}_j} = (x_{j,1}, x_{j,2})$, means $\overrightarrow{\mu_i} = (\mu_{i,1}, \mu_{i,2})^T$ and $\overrightarrow{\mu_j} = (\mu_{j,1}, \mu_{j,2})^T$ and diagonal covariance matrices $\Sigma_i = diag(\sigma_{i,1}^2, \sigma_{i,2}^2)$ and $\Sigma_j = diag(\sigma_{j,1}^2, \sigma_{j,2}^2)$ respectively. The difference between normal random vectors of O_i and O_j is given by $\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j} \sim \mathcal{N}(\overrightarrow{\mu_{i-j}}, \Sigma_{i-j})$, where $\overrightarrow{\mu_{i-j}} = \mu_i - \mu_j$ and $\Sigma_{i-j} = \Sigma_i + \Sigma_j$ [1].

Since Σ_i and Σ_j are diagonal matrices, the distribution functions are independent in coordinates. Hence the 2-dimensional normal difference distribution of uncertain objects O_i and O_j is given by,

$$
\begin{aligned}
f(\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}) =& \frac{1}{2\pi\sqrt{(\sigma_{i,1}^2 + \sigma_{j,1}^2)(\sigma_{i,2}^2 + \sigma_{j,2}^2)}} \\
& exp\left\{ -\left(\frac{(x - \alpha_1)^2}{(\sigma_{i,1}^2 + \sigma_{j,1}^2)} + \frac{(y - \alpha_2)^2}{(\sigma_{i,2}^2 + \sigma_{j,2}^2)} \right) \right\},
\end{aligned}
\tag{5}
$$

where $\alpha_1 = \mu_{i,1} - \mu_{j,1}$ and $\alpha_2 = \mu_{i,2} - \mu_{j,2}$ are the differences between the means of objects O_i and O_j and $\sigma_{i,1}^2, \sigma_{j,1}^2, \sigma_{i,2}^2$ and $\sigma_{j,2}^2$ are the variances of the uncertain objects O_i and O_j in dimensions 1 and 2 respectively.

Hence the probability that the uncertain object O_i lies within d-distance of uncertain object O_j is given by,

$$
\begin{aligned}
Pr(|\overrightarrow{\mathcal{A}_i} - \overrightarrow{\mathcal{A}_j}| \leq d) =& \frac{1}{2\pi\sqrt{(\sigma_{i,1}^2 + \sigma_{j,1}^2)(\sigma_{i,2}^2 + \sigma_{j,2}^2)}} \\
& \int_0^d \int_0^{2\pi} exp\left\{ -\left(\frac{(r\cos\theta - \alpha_1)^2}{2(\sigma_{i,1}^2 + \sigma_{j,1}^2)} + \frac{(r\sin\theta - \alpha_2)^2}{2(\sigma_{i,2}^2 + \sigma_{j,2}^2)} \right) \right\} r\, d\theta\, dr
\end{aligned}
\tag{6}
$$

■

Extracting Keyphrase Set with High Diversity and Coverage Using Structural SVM

Weijian Ni, Tong Liu, and Qingtian Zeng⋆

Shandong University of Science and Technology
Qingdao, Shandong, 266510 P.R. China
niweijian@gmail.com, liu_tongtong@foxmail.com,
qtzeng@163.com

Abstract. Keyphrase extraction plays an important role in automatic document understanding. In order to obtain concise and comprehensive information about the content of document, the keyphrases extracted from a given document should meet two requirements. First, the keyphrases should be diverse to each other so as to avoid carrying duplicated information. Second, every keyphrases should cover various aspects of the topics in the document so as to avoid unnecessary information loss. In this paper, we address the issue of automatic keyphrases extraction, giving the emphasis on the diversity and coverage of keyphrases which is generally ignored in most conventional keyphrase extraction approaches. Specifically, the issue is formulated as a subset learning problem in the framework of structural learning and structural SVM is employed to preform the task. Experiments on a scientific literature dataset show that our approach outperforms several state-of-the-art keyphrase extraction approaches, which verifies the benefits of explicit diversity and coverage enhancement.

1 Introduction

Keyphrases of a given document often refer to a number of phrases which are able to capture the main topics of that document. Due to the briefness and informativeness, keyphrases could make readers quickly obtain a rough understanding of document without going through the text. Besides, keyphrases have been utilized in a variety of natural language processing tasks such as digital library [1] and search engine [2].

Actually, authors are not always required to manually provide keyphrases. Therefore it would be helpful to automatically extracting keyphrases from documents, which is the aim of keyphrase extraction. Generally speaking, the task of keyphrase extraction consists of two stages: candidate phrase generation and keyphrase selection. In the first stage, the document is segmented into phrases, each of which is viewed as a keyphrase candidate. In the second but the key stage, each of the candidate phrases is assigned with a saliency score which is often calculated by leveraging supervised or unsupervised learning techniques, to

---

⋆ Corresponding author.

indicate whether it could be a keyphrase or not. In most conventional keyphrase extraction approaches, the candidates with the top-k scores will be selected as the keyphrases individually.

However, we observe that the saliency scores of semantically or lexically similar candidates also tend to be similar with each other, which may give rise to undesirable extracted results for conventional approaches. Take this paper for example, *keyphrase extraction* and *structural learning* are the desirable keyphrases. However, the candidates like *extracting keyphrases*, *keyphrase extraction approach* may share similar saliency scores with the true keyphrase *keyphrase extraction* for their lexical similarity. That would bring another true keyphrase *structural learning* down from the top ranked list of candidates from which the keyphrases are selected, if its saliency score was a bit lower than that of *keyphrase extraction*. Obviously, the keyphrase set of *keyphrase extraction* and *structural learning* is preferred to that of *keyphrase extraction*, *extracting keyphrases* and *keyphrase extraction approach*, as the former covers more aspects of the topics in this document while being more diverse to carry less redundant information.

In this paper, we propose a novel keyphrase extraction approach that explore the diversity and coverage characteristics of keyphrases extracted from a given document. The main idea is to regard the task of keyphrase extraction as a problem of subset learning where the keyphrases of a given document is a subset of the set of candidate phrases generated from that document. We formulate subset learning problem in the framework of structural learning and employ a large margin structural learning algorithm, i.e., structural SVM, with a specific loss function to learn extraction model. We evaluate our approach using a dataset composed of research articles with manually assigned keyphrases by the authors. The experimental results show the improvement of our approach to several state-of-the-art keyphrase extraction approaches, which results from the explicit enhancement of the diversity and coverage of keyphrase set.

2   Related Work

2.1   Keyphrase Extraction

Most recent studies have investigated the problem of keyphrase extraction from a machine learning point of view, which can be categorized into two major approaches: supervised and unsupervised.

Supervised approaches usually involve building a binary classifier using a collection of documents with manually assigned keyphrases as training set. The accuracy of the classifier relies heavily on the features describing the saliency of candidate phrase. Turney [3] calculated nine features such as the phrase frequency, the relative position of the phrase in document to represent each candidate phrase. More types of features have been explored, including linguistic knowledge such as PoS tag patterns [4], semantic information gleaned from a domain-specific thesaurus [5], and etc. Rather than employing classification methods, Jiang et al. [6] applied another machine learning technique, namely learning to rank to keyphrase extraction. Supervised approaches have been applied to extract keyphrases from

many particular types of documents such as product landing pages [7] and social snippets [8].

Unsupervised approaches calculate the saliency score of each keyphrase candidates mainly based on the co-occurrence characteristics of words appearing in the source documents. Mihalcea et al. [9] proposed a graph-based ranking model, referred to as TextRank, for keyphrase extraction. Recently, a number of extensions of graph-based methods have been proposed. For example, Liu et al. [10] calculated the saliency score of each word w.r.t various topics via multiple random walks; Wan et al. [11] built three types of graphs to reflect a variety of relationships between sentences and words. Clustering techniques have also been used in keyphrase extraction [12].

Our approach falls into the first category, but neither of the previous work explicitly takes the diversity and coverage of keyphrases into account which are key requirements of desirable keyphrases.

2.2 Structural SVM

Structural SVM [13] is a large margin approach to the problem of learning with structured and interdependent output space. It first extracts combined features from the input and output space, then learn a discriminant function using a generalized maximum-margin principle to derive prediction for a given input. Recently, some improvements have been conducted. For example, Joachims et al. [14] introduced a 1-slack version of the cutting plane algorithm with the time complexity linear in the number of training examples. Yu et al. [15] used approximate cutting planes and random sampling to enable efficient training structural SVM with kernels. Sarawagi et al. [16] proposed an improved training algorithm that generates more informative violated constraints during learning process. Recently, structural SVM has found increasing applications such as document retrieval [17], visual object localization [18]. To the best of our knowledge, it was not applied to the task of keyphrase extraction.

3 Keyphrase Set Extraction Approach

3.1 Formulations

Let $\mathbf{x} = \{x_1, \cdots, x_m\} \in \mathcal{X}$ denote a given document composed of m candidate phrases x_i, $\mathbf{y} \in \mathcal{Y}$ denote a subset of phrases selected as the keyphrases. As in supervised learning scenario, we are given a sample of input (document) and output (keyphrase set) pairs $S = \{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y} : i = 1, \cdots, n\}$ as training set, and aim to find a hypothesis function $h : \mathcal{X} \to \mathcal{Y}$ that minimize the empirical risk:

$$R_S^{\Delta}(h) = \frac{1}{n} \sum_{i=1}^{n} \Delta(\mathbf{y}_i, h(\mathbf{x}_i))$$

where $\Delta(\mathbf{y}_i, \tilde{\mathbf{y}})$ is a loss function that quantifies how well the predicted keyphrase set $\tilde{\mathbf{y}}$ approaches the ground truth $\mathbf{y}_i$. The loss function for keyphrase extraction will be described in detail in Sect. 3.3.

In the framework of structural learning, a linear discriminant function f : $\mathcal{X} \times \mathcal{Y} \to \mathbf{R}$ is employed to measure the compatibility of a predicted keyphrase set with the given document. Thus, the hypothesis function h takes the form of

$$h(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} f(\mathbf{x}, \mathbf{y}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle \tag{1}$$

where $\mathbf{w}$ is a parameter vector, and $\Psi(\mathbf{x}, \mathbf{y})$ is a combined feature representation of document and the corresponding predicted keyphrase set, which will be described in detail in Sect. 3.4.

Using the structural SVM, the parameter vector $\mathbf{w}$ is learned through optimizing the following convex quadratic programming problem:

$$\max_{\mathbf{w}; \xi_1, \cdots, \xi_n} \quad \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \quad \forall i, \forall \mathbf{y} \in \mathcal{Y} \backslash \mathbf{y}_i : \langle \mathbf{w}, \Psi(\mathbf{x}_i, \mathbf{y}_i) \rangle - \langle \mathbf{w}, \Psi(\mathbf{x}_i, \mathbf{y}) \rangle \geq 1 - \frac{\xi_i}{\Delta(\mathbf{y}_i, \mathbf{y})} \tag{2}$$

The object of the QP problem is a tradeoff between the structural risk $\frac{1}{2}\|\mathbf{w}\|^2$ and an upper bound of empirical risk, which is controlled by a parameter C. The constraints imply that the re-scaled margin between the ground truth $\mathbf{y}_i$ and the best runner-up $\tilde{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{Y} \backslash \mathbf{y}_i} \langle \mathbf{w}, \Psi(\mathbf{x}_i, \mathbf{y}) \rangle$ is maximized. As it is known that there are 2^n subsets of a set of size n, the number of constraints in (2) is intractably large. In the paper, we employ cutting plane algorithm to give a approximate solution of which the convergence can be theoretically guaranteed. For details of the algorithm please refer to [13].

After the QP problem in (2) is solved, the keyphrase set of a new document can be predicted using (1). Since exhaustive search over $\mathcal{Y}$ is NP-hard, motivated by the algorithm in [17], we propose a greedy algorithm to derive an approximate optimal keyphrase set for a given document.

3.2   Diversity Measure

The element of our approach to enhancing diversity and coverage of keyphrase set is to define a measure to quantify the diversity between keyphrases.

The existing similarity measures such as knowledge-based measures [19], corpus-based measures [20], web-based measures [21], give evidences of the similarity between two pieces of text from different points of view. However, neither of them is proposed for the application of measuring the topical diversity within a phrase set. It is thus not appropriate to utilize any single existing measure in our approach directly. In the paper, we take the existing similarity measures as meta measures and define the diversity measure between keyphrases as weighted combination of the meta measures. Based on the assumption that the keyphrases of a given document have large diversity, the combination weights are obtained supervised by a set of given keyphrase set as training set.

Formulations of Diversity Measure Learning. Let $f(x_1, x_2) \in [0,1]$ denote a function measuring the dissimilarity between two phrases x_1 and x_2. Given v off-the-shelf dissimilarity functions $F = \{f_1, \cdots, f_v\}$ as meta measures, and a training set $T = \{\mathbf{s}_1, \cdots, \mathbf{s}_r\}$, where $\mathbf{s}_i = \{x_1, \cdots, x_{n_i}\}$ $(i = 1, \cdots, n)$ is the keyphrase set of the i-th document and x_j $(j = 1, \cdots, n_i)$ is one of the keyphrases, the aim is to find a normalized weight vector $\mathbf{u} = \{u_1, \cdots, u_v\}$ which maximizes the diversities between all keyphrases in the training set:

$$\max_{\mathbf{u}} \sum_{i=1}^{r} \left(\frac{n_i(n_i - 1)}{2} \sum_{j=1}^{n_i} \sum_{k=j+1}^{n_i} \sum_{l=1}^{v} u_l f_l(x_{ij}, x_{ik}) \right)$$

$$\text{s.t.} \sum_{l=1}^{v} u_l = 1, \quad u_l \geq 0 \text{ for } l = 1, \cdots, v \tag{3}$$

Meta Measures. In the paper, we calculate the following four different types of dissimilarity measures between phrases, which are used as the meta measures.

1. *Lexical-based measure.* The measure basically counts the commonly appearing words in both phrases. We match the stemmed words instead of original words to enlarge the coverage of the measure.
2. *Knowledge-based measure.* WordNet [22] is used as external semantic knowledge source for the measure. We represent each word in the phrases by its corresponding synset in WordNet and count the commonly appearing synsets in both phrases.
3. *Corpus-based Measure.* Given a corpus represented by a word-by-document matrix, we derive the "concept space" of the corpus using LSA [23]. The diversity measure between phrases is then computed as the inner-product of "concept" vector representations of the phrases.
4. *Web-based measure.* The main idea of the measure is to leverage web search results to enrich the representation of short texts. We implement the web-base measure following [21].

Learning Algorithm. We employ a hill climbing algorithm for solving the optimization problem in (3). Algorithm 1 describes the details of the algorithm.

During each iterations, we search for a weight for each meta measure while persevering that of others and update only one weight. This helps the algorithm partially avoid overfitting from which conventional hill climbing algorithms often suffer. Besides, we run the algorithm with k random initializations and select the final weight vector with the maximum target value of (3) from the k runs.

3.3   Loss Function

In order to enhance diversity and coverage, we define the loss function in Structural SVM as the penalty of extraction errors w.r.t. diversity and coverage. Intuitively, the diversity error of a predicted keyphrase set refers to extracting

Algorithm 1. Hill climbing algorithm for diversity measure learning

1: **Input:** $T = \{\{x_{11}, \cdots, x_{1n_1}\}, \cdots, \{x_{r1}, \cdots, x_{rn_r}\}\}$, $F = \{f_1, \cdots, f_v\}$, $\epsilon > 0$

2: Initialize $(u_1, \cdots, u_v)$ as a random vector satisfying the constraints in (3)

3: $D_{T,F}(u_1, \cdots, u_v) = \frac{1}{r}\sum_{i=1}^{r}\left(\frac{n_i(n_i-1)}{2}\sum_{j=1}^{n_i}\sum_{k=j+1}^{n_i}\sum_{l=1}^{v}u_l f_l(x_{ij}, x_{ik})\right)$

4: $max = max_{\text{for}} = D_{T,F}(u_1, \cdots, u_v)$

5: **repeat**

6:   **for** $l = 1$ **to** $v - 1$ **do**

7:     $temp = u_l$

8:     $u'_l = \arg\max_s D_{T,F}(u_1, \cdots, u_{l-1}, s, u_{l+1}, \cdots, u_v)$ $(u_v = 1 - \sum_{i=1}^{v-1}u_i)$

9:     $max' = D_{T,F}(u_1, \cdots, u_{l-1}, u'_l, u_{l+1}, \cdots, u_v)$ $(u_v = 1 - \sum_{i=1}^{v-1}u_i)$

10:     **if** $max' > max_{\text{for}}$ **then**

11:       $(max_{\text{for}}, idx, val) = (max', l, u'_l)$

12:     **end if**

13:     $u_l = temp$

14:   **end for**

15:   **if** $max_{\text{for}} - max < \epsilon$ **then**

16:     **break**

17:   **end if**

18:   $(max, u_{idx}) = (max_{\text{for}}, val)$

19: **until true**

20: **Output:** $(u_1, \cdots, u_v)$

topically similar phrases as the keyphrases, and the coverage error refers to missing important topics in the extracted results. Figure 1 illustrates the two types of errors on a keyphrase set consists of three keyphrases. Figure 1(a) shows the ideal case that the extracted keyphrases (denoted by circles) are identical to the true keyphrases (denoted by triangles). Figure 1(b) shows a mis-extracted case that two similar phrases $1'$ and $3'$ are extracted on account of a true keyphrase 1, while another true keyphrase 3 is missing in the extracted set. In the paper, we simply assume the topics are delivered by keyphrases. Therefore it can be regarded that both of the diversity and coverage errors occurs in Fig. 1(b).

Let $\mathbf{y} = \{y_1, \cdots, y_k\}$ and $\tilde{\mathbf{y}} = \{\tilde{y}_1, \cdots, \tilde{y}_l\}$ denote the true keyphrase set and the predicted keyphrases of a document, respectively. Before defining the loss function, we first obtain a partition of $\tilde{\mathbf{y}}$ according to the similarities between the predicted keyphrases and the ground truth, i.e.,

$$\tilde{\mathbf{y}} = \tilde{\mathbf{y}}(y_1) \cup \cdots \cup \tilde{\mathbf{y}}(y_k), \quad \forall\, 1 \leq i, j \leq k,\ \tilde{\mathbf{y}}(y_i) \cap \tilde{\mathbf{y}}(y_j) = \varnothing \qquad (4)$$

where $\tilde{\mathbf{y}}(y_i) = \{\tilde{y}_j \,|\, \tilde{y}_j \in \tilde{\mathbf{y}},\ y_i = \arg\min_{y_h \in \mathbf{y}} d(\tilde{y}_j, y_h)\}$ is the predicted keyphrases on account of the true keyphrase y_i, and $d(\cdot, \cdot) \in [0, 1]$ is the diversity measure learned through Algorithm 1. Note that $\tilde{\mathbf{y}}(y_i)$ may consist of more than one phrases or be null, if the diversity and coverage errors occur on y_i, respectively.

Using the reorganized form of predicted keyphrases in (4), the loss function $\Delta(\mathbf{y}, \tilde{\mathbf{y}})$ is defined as follows:

$$\Delta(\mathbf{y}, \tilde{\mathbf{y}}) = \frac{1}{|\mathbf{y}|}\sum_{y_i \in \mathbf{y}}\sum_{\tilde{y}_j \in \tilde{\mathbf{y}}(y_i)}\frac{d(y_i, \tilde{y}_j)}{|\tilde{\mathbf{y}}(y_i)|} \qquad (5)$$

(a) Ideal case   (b) A mis-extraction case
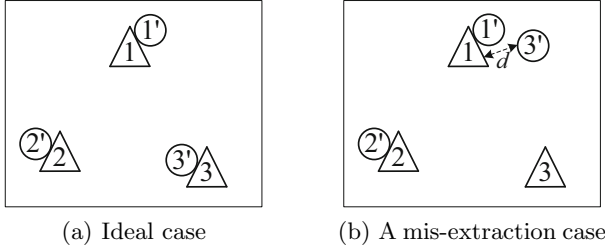
Fig. 1. Illustration of keyphrase set extraction errors

As for the above loss function, we can easily find that:

1. $\Delta(\mathbf{y}, \tilde{\mathbf{y}}) = 0$ if and only if $\tilde{\mathbf{y}}(y_i) = \{y_i\}$ holds for each $y_i \in \mathbf{y}$.
2. $\Delta(\mathbf{y}, \tilde{\mathbf{y}}) = 1$ if and only if $d(y_i, \tilde{y}_j) = 1$ holds for each $y_i \in \mathbf{y}$ and $\tilde{y}_j \in \tilde{\mathbf{y}}$.
3. If no phrases is extracted on account of a true keyphrase y_i, i.e., $|\tilde{\mathbf{y}}(y_i)| = 0$, the loss w.r.t. y_i is $1/|\mathbf{y}|$.
4. If more than one phrases is extracted on account of a true keyphrase y_i, i.e., $|\tilde{\mathbf{y}}(y_i)| > 1$, the loss w.r.t. each $\tilde{y}_j \in \tilde{\mathbf{y}}(y_i)$ is the diversity between $\tilde{y}_j$ and y_i scaled by $1/|\tilde{\mathbf{y}}(y_i)|$.

3.4 Features

Our approach use a variety of features to represent each possible keyphrase sets rather than possible keyphrases. Given an input (document) and output (keyphrase set) pair $(\mathbf{x}, \mathbf{y})$ where $\mathbf{y} = \{y_1, \cdots, y_k\}$, we first extract a set of features $\psi_p(\mathbf{x}, y_i)$ $(p = 1, \cdots, d)$ from each keyphrase candidates $y_i \in \mathbf{y}$, referred to as basic features, then compute the feature $\Psi_{p,op}(\mathbf{x}, \mathbf{y})$ on the basis of $\psi_p(\mathbf{x}, y_i)$ $(i = 1, \cdots, k)$ by using a predefined feature operator op. Finally, all the features $\Psi_{p,op}(\mathbf{x}, \mathbf{y})$ are concatenated to form the feature vector $\Psi(\mathbf{x}, \mathbf{y})$.

Basic Features. We make use of the following features widely used in conventional keyphrase extraction approaches as the basic features in our approach.

1. *TF-IDF.* Phrase and word level TF, IDF and TF×IDF are computed as features. Since a phrase may contain more than one words, we make use of the average, maximum and minimum of word level values.
2. *Phrase length.* The features includes the numbers of words and letters in the phrase as well as the length values binned by several length levels.
3. *Occurrence.* We compute binary features indicating whether the phrase occurs in certain part of the document. The first occurrence and the distribution of the phase in the document are computed as real-valued features.
4. *Part of speech.* We compute binary features indicating whether the phrase starts with, ends with or contains a certain part of speech.
5. *PageRank.* Phrase and word level PageRank values are computed as features. Just as the TF-IDF features, the average, maximum and minimum of word level values are taken as features.

Feature Operators. After extracting a set of basic features on every phrases in a phrase set, a number of feature operators are defined to compute the feature values of the keyphrase set. Each of them takes a set of values of a basic feature as input, and output a feature value of the corresponding keyphrase set. The feature operators used in our approach are listed in Table 1.

Table 1. Description of feature operators

Operator	Description	Example	
		Input	Output
1 max	Maximum of a set of values	$\{1.1, 1.5, 9\}$	9
2 min	Minimum of a set of values	$\{1.1, 1.5, 9\}$	1.1
3 avg	Average of a set of values	$\{1.1, 1.5, 9\}$	3.9
4 $stdev$	Standard deviation of a set of values	$\{1.1, 1.5, 9\}$	4.5
5 all	Are all the values in the set equal to 1?	$\{1, 1, 0\}$	0
6 $none$	Are all the values in the set equal to 0?	$\{1, 1, 0\}$	0
7 one	Is at least one value in the set equal to 1?	$\{1, 1, 0\}$	1
8 $half$	Are more than half values in the set equal to 1?	$\{1, 1, 0\}$	1

Note that the operators in Table 1 would be applied to different types of basic features. Specifically, the first four operators are applied to real-valued basic features and the others are applied to binary ones.

4   Experiments

4.1   Experimental Setup

Dataset. To avoid manually annotation of keyphrases which is often laborious and erroneous, we constructed an evaluation dataset using research articles with author provided keyphrases. Specifically, we collected the full-text of papers published in the proceedings of two conferences, namely ACM SIGIR and SIGKDD from 2006 to 2010. After removing the papers without author provided keyphrases, there are totally 3461 keyphrases appearing in 997 papers in our evaluation dataset. For each paper, tokenization, pos tagging, stemming and chunking were performed using NLTK (Natural Language Toolkit). We observed that the keyphrases make up only 0.31% of the total phrases in the dataset, i.e., the dataset is highly imbalanced. To compensate for this, we filtered out a portion of negative instances (non-keyphrases) from the dataset using some heuristic rules.

Baselines. Two state-of-the-art unsupervised and supervised keyphrase extraction approaches, namely TextRank [9] and Kea [24] are selected as the baselines. We also compared our approach with SVM that made use of all the basic features introduced in Sect 3.4 for training. The SVM baseline and our approach

are implemented using the SVM^{light} toolkit[1]. As for the parameters C in SVM and SVM-struct, the default values given by SVM^{light} are used.

Evaluation Measures. The traditional metrics, namely *Precision*, *Recall* and *F1-score* are utilized to evaluate our approach and the baselines in the experiments.

4.2 Experimental Results

We random split the dataset into five even parts and conducted 5-fold cross evaluation. The extraction results of our approaches and the baselines averaged over the 5 trails are shown in Fig. 2.
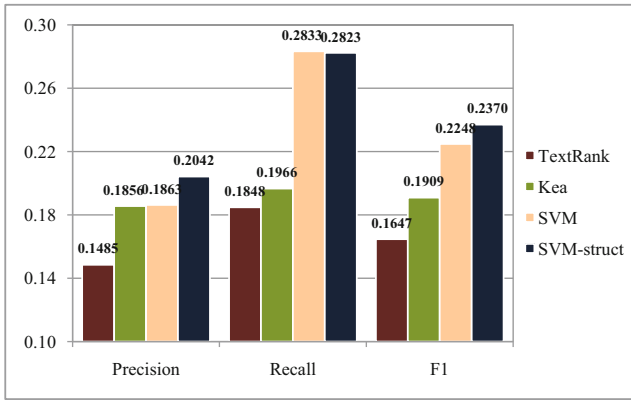
Fig. 2. Keyphrase extraction accuracies on scientific literature dataset

From the figure, we can see that our approach (denoted by SVM-struct) outperforms nearly all the baselines in terms of Precision, Recall and F1-score. We also conducted significance test (paired t-test) on the differences between our approach and the baselines. The results show that the improvement of our approach over SVM in terms of Precision and F1-score is statistically significant ($p \leq 0.05$) and the improvement of our approach over TestRank and Kea in terms of all measures is statistically significant ($p \leq 0.05$). Since our approach uses the same set of basic features as does SVM , we can conclude that structural learning is benifical to the task of keyphrase extraction.

4.3 Analyses

Experimental analyses were conducted to give better understandings of the effectiveness of our approach.

---

[1] svmlight.joachims.org

Size of Keyphrase Set. The size of predicted keyphrase set specified in the greedy algorithm for keyphrase set prediction using (1) is the main parameter that may influence the performance of our approach. We ranged the parameter from 2 to 10 in our experiments and plotted the performance variation of the our approach together with that of the SVM baseline in Fig. 3.
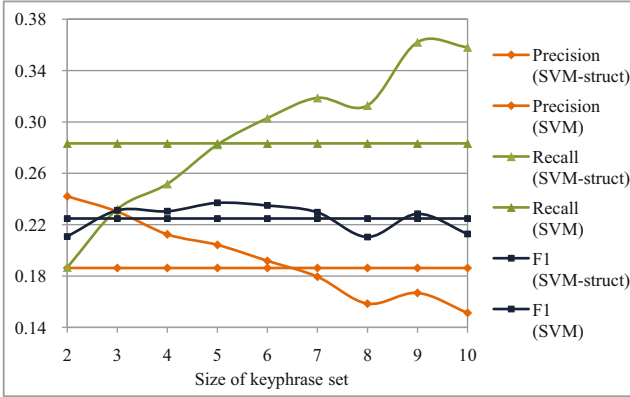
Fig. 3. Keyphrase extraction accuracies when varying the size of keyphrase set

From the figure, we can see that our approach outperforms SVM in terms of F1-score when the size equals $3, 4, 5, 6, 7, 9$. Since there are average 4 keyphrases in each document in our evaluation set, it can be concluded that the parameter of the size of keyphrase set could be reliably set to be slightly higher than the general average number of keyphrases per document in the target domain.

Diversity of Extraction Results. We investigated the ability of our approach to enhance the diversity and coverage of the extracted keyphrase sets. Given a set of documents with corresponding extracted keyphrase sets $S = \{(\mathbf{x}_1, \tilde{\mathbf{y}}_1), \cdots, (\mathbf{x}_n, \tilde{\mathbf{y}}_n)\}$, we computed the diversity of the extracted keyphrase sets as follows:

$$d(S) = \frac{1}{n} \sum_{i=1}^{n} \frac{n_i(n_i - 1)}{2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} d(\tilde{y}_{ij}, \tilde{y}_{ik})$$

where $d(\tilde{y}_{ij}, \tilde{y}_{ik})$ is the diversity between two phrases in an extracted keyphrase set computed using the diversity measure learned by Algorithm 1. We randomly sampled 100 documents from the evaluation set and compared the diversity of the extracted keyphrase sets of our approach with that of the ground truth and SVM baseline in Fig. 4.

From the figure, we can see that the average diversity of the extracted keyphrase sets by our approach is higher than that of the ground truth and SVM baseline. Besides, the diversity differences are all statistically significant
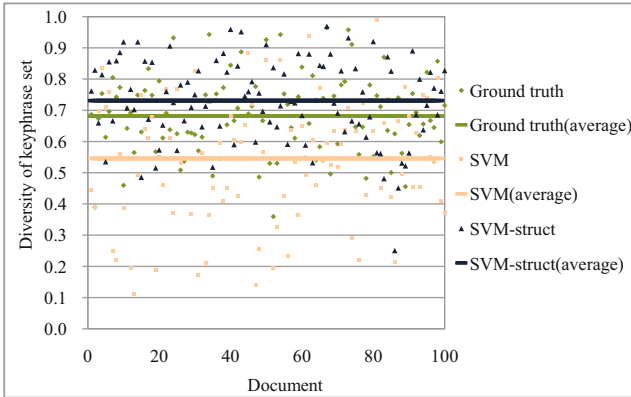
Fig. 4. Diversities of extracted keyphrase sets

in paired t-test ($p \leq 0.01$). The results indicate that our approach is capable of enhancing the diversity of extracted keyphrase set. It is also not difficult to conclude that the coverage of extracted keyphrase set can be enhanced resulting from the high diversity between extracted keyphrases and a proper setting of parameter of the size of extracted keyphrase set.

5   Conclusion and Future Work

In this paper, we have proposed a novel keyphrase extraction approach that takes the diversity and coverage of the keyphrases into accounts. In our approach, the issues of keyphrase extraction was formulated as subset learning which can be viewed as a structural learning problem. We have defined a loss function to reflect the diversity and coverage characteristics of the keyphrase set and preformed training using structural SVM. Experimental results on a scientific literature dataset have shown the ability of our approach to enhancing diversity and coverage which results in the performance improvement to state-of-the-art approaches.

As future work, we will experimentally verify the effectiveness of our approach with more evaluation sets constructed from widely domains. In addition, we plan to leverage topic models such as LDA and LSA to derive latent topics in document so as to enhance the diversity and coverage of extracted keyphrases.

References

1. Lehtonen, M., Doucet, A.: Enhancing Keyword Search with a Keyphrase Index. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 65–70. Springer, Heidelberg (2009)
2. Wu, Y., Li, Q.: Document Keyphrases as Subject Metadata: Incorporating Document Key Concepts in Search Results. Information Retrieval 11, 229–249 (2008)
3. Turney, P.D.: Learning Algorithms for Keyphrase Extraction. Information Retrieval 2, 303–336 (2000)
4. Hulth, A.: Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In: Proceedings of EMNLP, pp. 216–223 (2003)
5. Medelyan, O., Witten, I.H.: Thesaurus Based Automatic Keyphrase Indexing. In: Proceedings of JCDL, pp. 296–297 (2006)
6. Jiang, X., Hu, Y., Li, H.: A Ranking Approach to Keyphrase Extraction. In: Proceedings of SIGIR, pp. 756–757 (2009)
7. Yih, W., Goodman, J., Carvalho, V.R.: Finding Advertising Keywords on Web Pages. In: Proceedings of WWW, pp. 213–222 (2006)
8. Li, Z., Zhou, D., Juan, Y., Han, J.: Keyword Extraction for Social Snippets. In: Proceedings of WWW, pp. 1143–1144 (2010)
9. Mihalcea, R., Tarau, P.: TextRank: Bringing Order into Texts. In: Proceedings of EMNLP, pp. 404–411 (2004)
10. Liu, Z., Huang, W., Zheng, Y., Sun, M.: Automatic Keyphrase Extraction via Topic Decomposition. In: Proceedings of EMNLP, pp. 366–376 (2010)
11. Wan, X., Yang, J., Xiao, J.: Towards an Iterative Reinforcement Approach for Simultaneous Document Summarization and Keyword Extraction. In: Proceedings of ACL, pp. 552–559 (2007)
12. Grineva, M., Grinev, M., Lizorkin, D.: Extracting Key Terms From Noisy and Multi-theme Documents. In: Proceedings of WWW, pp. 661–670 (2009)
13. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large Margin Methods for Structured and Interdependent Output Variables. JMLR, 1453–1484 (2005)
14. Joachims, T., Finley, T., Yu, C.J.: Cutting-plane training of structural SVMs. Machine Learning, 27–59 (2009)
15. Yu, C.J., Joachims, T.: Training Structural SVMs with Kernels Using Sampled Cuts. In: Proceeding of SIGKDD, pp. 794–802 (2008)
16. Sarawagi, S., Gupta, R.: Accurate Max-Margin Training for Structured Output Spaces. In: Proceedings of ICML, pp. 888–895 (2008)
17. Yue, Y., Joachims, T.: Predicting Diverse Subsets Using Structural SVMs. In: Proceedings of ICML, pp. 1224–1231 (2008)
18. Zhu, L., Chen, Y., Yuille, A., Freeman, W.: Latent Hierarchical Structural Learning for Object Detection. In: Proceedings of CVPR, pp. 1062–1069 (2010)
19. Wan, S., Angryk, R.A.: Measuring semantic similarity using wordnet-based context vectors. In: Proceedings of IEEE ICMSC, pp. 908–913 (2007)
20. Islam, A., Inkpen, D.: Semantic Text Similarity Using Corpus-Based Word Similarity and String Similarity. ACM TKDE 2, 10–25 (2008)
21. Sahami, M., Heilman, T.D.: A Web-based Kernel Function for Measuring the Similarity of Short Text Snippets. In: Proceedings of WWW, pp. 377–386 (2006)
22. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM 38, 39–41 (1995)
23. Landauer, T., Foltz, P.W., Laham, D.: Introduction to Latent Semantic Analysis. Discourse Processes 25, 259–284 (1998)
24. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C.: KEA: Practical Automatic Keyphrase Extraction. In: Proceedings of JCDL, pp. 254–255 (1999)

Discovering the Most Potential Stars in Social Networks with Infra-skyline Queries

Zhuo Peng, Chaokun Wang, Lu Han, Jingchao Hao, and Xiaoping Ou

School of Software, Tsinghua University, Beijing 100084, China
Tsinghua National Laboratory for Information Science and Technology
Key Laboratory for Information System Security, Ministry of Education
{pengz09,hanl09,oxp10,haojc10}@mails.tsinghua.edu.cn
chaokun@tsinghua.edu.cn

Abstract. With the rapid development of *Social Network* (SN for short), people increasingly pay attention to the importance of the roles which they play in the SNs. As is usually the case, the standard for measuring the importance of the members is multi-objective. The skyline operator is thus introduced to distinguish the important members from the entire community. For decision-making, people are interested in the most potential members which can be promoted into the skyline with minimum cost, namely the problem of *Member Promotion in Social Networks*. In this paper, we propose some interesting new concepts such as *Infra-Skyline* and *Promotion Boundary*, and then we exploit a novel promotion boundary based approach, i.e., the InfraSky algorithm. Extensive experiments on both real and synthetic datasets are conducted to show the effectiveness and efficiency of the InfraSky algorithm.

Keywords: SNA, Member Promotion, Infra-skyline, Promotion Boundary.

1 Motivation

As the online social communities become prevailing, more and more daily social activities now take place in Social Networks (SN for short) due to convenience and low cost. The importance of the role which every single member plays in the SNs is attracting more and more attention. Because those important members in the SNs, usually known as stars, e.g., well-known companies, popular actors, famous scholars, attract more public attentions, obtain more trusts, and also take more responsibilities. Under the circumstances, member promotion, which aims at finding the most potential star member(s) to perform promotions, has been introduced into the literature recently [11]. As a matter of fact, most promotions are multi-objective in real applications, e.g., both authoritative and active experts, creative, skilled and even experienced workers. The skyline operator has thus been introduced to measure the importance of the members in SNs in [11]. Provided that a point a in a set D is as good or better in all dimensions and better in at least one dimension than another point b in D, we say a dominates

b. The skyline of D is defined as those points that are not dominated by any other points in D. Therefore, the problem of member promotion in SNs is defined as to excavate the most appropriate non-skyline member(s), namely the most potential stars, which can be promoted into skyline at a minimum cost.

An SN is commonly modeled as a directed graph. Intuitively from the graph topology perspective, a member with many incoming edges embodies authoritativeness in the SN, while the member which owns a large number of outgoing edges serves as a hub member and helps constructing the network and spreading the information. It is simple but reasonable to utilize the indegree and outdegree as the dimensions for the skyline query. So in order to promote some specific members, we have to tune the indegrees and outdegrees. Intuitively, we employ edge addition as the promotion manner. Based on the assumption that only one edge is allowed to connect any two members in a specific direction, we try to add new edges to the target member to increase its indegree or outdegree or both simultaneously, and thus promote it into the skyline.

The contributions of this paper are summarized as follows. 1) We bring forward some new concepts such as *Infra-skyline* and *Promotion Boundary* for pruning the search space and accelerating the computation of the promotion costs. 2) Based on the concepts, we propose the InfraSky algorithm which promises high efficiency and minimum promotion cost. Detailed theoretical analyses are provided to prove the correctness and effectiveness of the algorithm. 3) We conduct extensive experiments to show the effectiveness and efficiency of the algorithm.

The remainder of this paper is organized as follows. Section 2 reviews related work. Then we introduce some basic concepts in Section 3. In Section 4, we bring forward a novel pruning strategy and a new manner for promotion cost computation based on *Infra-skyline* and *Promotion Boundary*, therefore propose the InfraSky algorithm. In Section 5, we show the results of our extensive experiments. Finally, we conclude our study and give some interesting directions for future study in Section 6.

2   Related Work

Given a set of D-dimensional data points, skyline query, also known as Pareto optimal problem or maximum vectors problem, retrieves a subset of data points that are not dominated by any other points [8]. It begins the skyline-related studies for data management as the skyline operator [1]. With the development of these studies, a series of algorithms which are not based on index have been proposed, e.g., Block-nested-loops [1], D&C [1]. By utilizing indices such as B^+-tree, R-tree are introduced into skyline query, the index-based approaches remarkably improve the efficiency, e.g., Index [13] based on B^+-tree, NN [7] based on R-tree, BBS [9] based on R-tree. As a multi-objective optimization tool, skyline has plenty of applications in many scenarios, e.g., queries in databases [1], [10], data mining [6], queries in metric spaces [2], [4], road networks [3], spatial spaces [12] and large graphs [15].

Promotion, as an important concept in marketing, has been introduced into data management applications recently. Wu et al. propose an approach to find

such appropriate subspaces that the target can be prominent in these subspaces [14]. Query for the points that can be changed to be a skyline point at the minimum cost is studied in [5]. The work only fits the case that the points are independent from each other, however, the change of the metrics of one member in the SN usually affects other members.

Member promotion in SNs is first proposed and studied in [11]. The formal definition is provided and a brute-force algorithm is proposed at first. Based on the characteristics of the skyline and the promotion process, several dominance-based optimization strategies are proposed to improve the efficiency, and thus leads to the IDP algorithm. However, the optimization strategies are not good enough. The dominance based dynamic pruning strategy costs much time to generate a list of skippable plans and still needs more time to query for a match during the process. And the effect of the pruning strategy is not significant. The dominance indexes will be changed frequently by edge additions during the promotion process, and thus will cause a number of computations. In the worst case when the dominance relationship between the members is rare in the network, the time cost will still be exponential.

3   Preliminaries

In this paper, an SN is modeled as a weighted directed graph $G(V, E, W)$. The elements in V represent the members in the SN. E includes the existing directed edges between the members. Each weight w in the weight matrix W is the cost for establishing the directed relation between any two different nodes. The weights are all positive and the cost for establishing a self loop on any member is considered as infinite. The algorithms in [11] are proposed to deal with the SNs in which the weights of the edges are arbitrary positive real numbers. In some cases, however, the weights can be ignored or considered as all equal, e.g., in a group in which everyone already knows each other, the cost for any two different persons to build a connection can be ignored, or a probable contagious virus carrier may randomly infect other people with the same possibility in a public place. In this paper we only consider the case that the weights are ignored or all equal only, which is actually a special case of arbitrary weights. For simplicity, all the weights are set to be 1.

Example 1. Figure 1 shows a sample SN with 10 members ($n_0 \sim n_9$). The solid directed lines represent the directed relations between the nodes. The dotted ones are some possible connections we can build to promote some members. Using indegree and outdegree of the members as the dimensions of the skyline operator, the skyline members (n_0, n_2, n_3, n_5) are the solid nodes shown in the figure based on the degree distribution given in Figure 2.

Given an SN $G(V, E, W)$, let S_G be the skyline of G, when $S_G \neq V$, we say node $c \in V - S_G$ is a candidate for member promotion.

Definition 1 (Promotion Plan). *Given an SN $G(V, E, W)$, against each candidate c, $p \subseteq V \times V$, we define p as a plan for c when p satisfies:*
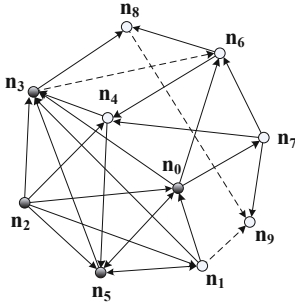
Fig. 1. A social network example

Fig. 2. Degree Distribution

	indegree	outdegree
n_0	3	4
n_1	2	3
n_2	0	5
n_3	5	1
n_4	3	2
n_5	4	3
n_6	2	2
n_7	1	3
n_8	2	0
n_9	1	0

(1) $p \subseteq \{e | (e = (c, \bullet) \vee e = (\bullet, c)) \wedge e \neq (c, c) \wedge e \notin E\}$,

(2) $c \in S_{G'}$, while $G' = (V, E + p, W)$.

In order to locate the members which can be promoted into the skyline with minimum cost by adding new plans, we need to define the cost model first.

Definition 2 (Promotion Cost). *Given an SN $G(V, E, W)$, for any candidate c, let p be a promotion plan for the promotion against c. The cost of promotion plan p, marked as $Cost(c, p)$, is the sum of the weights corresponding to the edges included in p. Assume the original vector composed by the indegree and outdegree of c is α, after promoted by p, it grows to β. Then under the situation of all weights set to be 1, the cost of p can be denoted as the L_1 distance between α and β, namely,*

$$Cost(c, p) = \mathrm{MD}(\alpha, \beta). \tag{1}$$

The function MD here serves as a calculator of the L_1 distance between two vectors. The promotion cost against c, marked as $\zeta(c)$, is the minimum cost among all the promotion plans which can get c promoted, namely,

$$\zeta(c) = min(Cost(c, p)). \tag{2}$$

Definition 3 (Infra-skyline). *Given an SN $G(V, E, W)$, the infra-skyline of G is the skyline of all the non-skyline members of G, namely, $I_G = S_{G-S_G}$, where G-S_G is a new SN generated from G by eliminating its skyline members.*

If we project all the nodes in G into a two-dimensional Cartesian coordinate system using the indegree as the x-axis and the outdegree as the y-axis. Let the projection be $m : V \rightarrow \mathrm{N}^2$, we project node v into point $m(v)$ in the coordinate system. As shown in Figure 3(a), after the projection, the skyline members are the black solid points, while the infra-skyline members are those dark gray solid points. By the way, when two members v_1, v_2 own the same indegree and outdegree, the two projective points will be coincident, namely, $m(v_1) = m(v_2)$.

(a) Infra-skyline

(b) Promotion Boundary
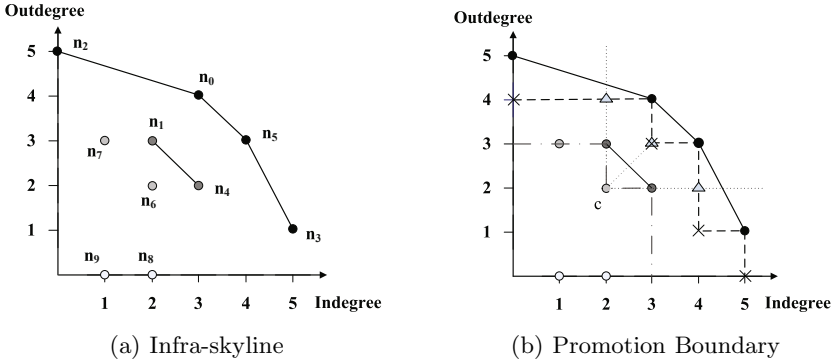
Fig. 3. Infra-skyline & Promotion Boundary

Definition 4 (Virtual Promotion Point). *Let K be the projective point set of the skyline members which contains no duplicates, namely $K = \{m(s)|s \in S_G\}$. Thus $K' = \{s_i| \ x(s_i) < x(s_{i+1}), \ x(s_i) \neq 0, \ y(s_i) \neq 0, \ s_i \in K, \ i = 1, 2, \cdots, f\}$ contains the points, which belong to K but not on the axes, ascending sorted by x coordinate. There will be $|K'|+1$ virtual promotion points denoted as*

$$t_i = \begin{cases} (0, y(s_i)) & i = 0 \\ (x(s_i), y(s_{i+1})) & i = 1, 2, \cdots, |K'| - 1 \ , \\ (x(s_f), 0) & i = |K'| \end{cases} \tag{3}$$

where $s_i \in K'$, $i = 1, 2, \cdots, |K'|$, and we mark the whole set as VP_G.

Definition 5 (Promotion Boundary). *Given an SN $G(V, E, W)$ and K', let b be the mark of a line segment between a skyline projective point s_i and its neighbor virtual promotion point t_{i-1} or t_i, namely line segment $t_{i-1}s_i$ or s_it_i, the promotion boundary of the skyline B_{S_G} can be presented as:*

$$B_{S_G} = \{b|b = t_{i-1}s_i \vee b = s_it_i\} \tag{4}$$

where $t_i \in VP_G$, $s_i \in S_G$, $i = 0, 1, \cdots, |K'| - 1$.

4   Algorithm

Intuitively, we can employ the brute-force algorithm and by far the best algorithm proposed in [11], namely the index-based dynamic pruning (IDP) algorithm, to solve the problem because the equal-weighted SN is a special case of arbitrary-weighted SN. To be specific, we use a hierarchy traversal to verify all the possible promotion plans in an ascending order of the size against each candidate before we get the optimal result in the brute-force algorithm. However, due to the exponential number of possible plans in the search space, most of which are "meaningless", the time cost of this brute-force algorithm is exponential. We can then use the IDP algorithm which improves the efficiency based on

some dominance-based optimization strategies at both pruning stage and result checking stages. Nevertheless, in the worst case when the dominance relationship between the members is rare in the network, the effect of the optimization strategies will be weakened. As a result, the time complexities of both algorithms are beyond tolerance, especially when it encounters large sparse networks. In this paper, based on the promotion boundary and the infra-skyline, we find the promotion process can be notably improved in both candidate filtering and cost calculation. Thus we propose a novel approach, i.e. the InfraSky algorithm, to solve the problem effectively.

4.1   InfraSky

After we project all the nodes in G into the coordinate system using projection m, the projective point of the skyline members can be located in the right top contour. From the geometric point of view, the promotion process is just the process of shifting the projective point of the candidate to the contour. As a result, the problem can be described as to find out those members whose projective point can shift to the contour and finally get promoted successfully at the minimum L_1 distance. Intuitively, the algorithm can be obviously optimized on the basis of promotion boundary.

First of all, we need to present two observations on the relationship between promotion boundary and the promotion process.

Observation 1. *Given an SN $G(V, E, W)$, for any candidate c, let p be a plan for the promotion against c. Let the members in S_G which dominate c stay the same during the whole promotion process. We say c is successfully promoted iff $m(c)$ exceeds B_{S_G} or coincides with one of the projective points of the skyline members, which also belong to B_{S_G}.*

However, it is possible that S_G will change during the promotion process. But against any candidate c, we just need to ensure those new members or the updated members in S_G do not dominate c, and then the result of promotion stays the same. If a dominator of c is unavoidably generated, the plan is not able to promote c successfully, we have to add more edges.

Observation 2. *For any candidate c, $\zeta(c) = min(d_x, d_y, d_i, d_j)$. There must be at least one promotion plan p^*, the cost of which is the promotion cost of c. Namely, $Cost(c, p) = \zeta(c)$. d_x, d_y, d_i, d_j are calculated as follows.*

1) *$d_x = \text{MD}(m(c), p_x) + 1$, $(p_x(x_1, y_1) \in B_{S_G}) \wedge (p_x \in l_x)$, in which ray l_x is emitted from $m(c)$ and parallel to the positive direction of the x-axis.*
2) *$d_y = \text{MD}(m(c), p_y) + 1$, $(p_y(x_2, y_2) \in B_{S_G}) \wedge (p_x \in l_y)$, in which ray l_y is emitted from $m(c)$ and parallel to the positive direction of the y-axis.*
3) *$d_i = \text{MD}(m(c), vp_i) + 2$ $(i = 1, 2, \cdots, m)$, where vp_i is a virtual promotion point which dominates c.*
4) *$d_j = \text{MD}(m(c), m(s_j))(j = 1, 2, \cdots, n)$, where $s_j \in S_G$, and s_j dominates c.*

Algorithm 1. *Promote*(*candidate*)

Input   : *candidate* : a candidate for promotion
Output : promotion cost of *candidate* and the corresponding promotion plans

1 $cost = $ MAX;
   /* get the intersection of B_{S_G} and l_x, l_y */
2 $p_x = getintersection(l_x, B_{S_G})$;
3 $p_y = getintersection(l_y, B_{S_G})$;
   /* get the virtual promotion points set */
4 $\Gamma = getvpps(l_x, l_y, B_{S_G})$;
   /* get the skyline members which dominate *candidate* */
5 $\Lambda = getsps(l_x, l_y, B_{S_G})$;
6 $cost = min(cost, \text{MD}(candidate, p_x))$;
7 $cost = min(cost, \text{MD}(candidate, p_y))$;
8 **for** each $vp_i \in \Gamma$ **do**
9 $\quad$ $cost = min(cost, \text{MD}(candidate, vp_i))$;
10 **for** each $s_j \in \Lambda$ **do**
11 $\quad$ $cost = min(cost, \text{MD}(candidate, s_j))$;
   /* try plans with more edges once new dominators unavoidably
      generated by all plans with $cost$ edges */
12 **while** *verify*($cost$) is false **do**
13 $\quad$ $cost + +$;
14 *store the promotion plans*;
15 **return** $mCost$ and corresponding promotion plans

Based on Observation 1, we have already known that the promotion plan
with the cost lower than $\zeta(c)$ does not exist. Assume that all the costs of the
promotion plans for c are higher than $\zeta(c)$, which means the count of the edges
contained in each promotion plan should be more than $\zeta(c)$. Take $\zeta(c) = d_x$ for
example, there must be at least one promotion plan p whose cost is d_x can make
$m(c)$ exceed B_{S_G} or coincide with one of the projective points corresponding to
the skyline members. Nevertheless, promotion plans like p are not able to get
c promoted, which is an obvious contradiction with Observation 1. The other
cases can be verified similarly. The assumption does not hold.

Algorithm 1 presents the method to find the promotion cost and corresponding
promotion plans for any candidate based on the observations. Apparently, the
members which require larger L_1 distance can be pruned before the promotion.
From the geometric point of view, the L_1 distances between the members in B_{I_G}
and those in S_G are no larger than the distances between the other members and
those in S_G. According to the observations and the characteristics of skyline, we
can prune the candidate set to C_G based on the theorem below.

Theorem 1. *Given an SN $G(V, E, W)$, any optimal candidate c must belong
to C_G, where $C_G = I_G \bigcup \{c|\ c \notin S_G \wedge m(c) \in B_{I_G}\}$.*

Algorithm 2. *MemberPromotion*($\mathcal{G}$)

Input   : $\mathcal{G}$: weighted directed graph, $\mathcal{G}(V, E, W)$
Output : optimal members and the corresponding promotion plans of the
           minimum cost

```
/* initialize the global minimum cost */
```
1 $minCost = \text{MAX}$;
2 $S = skylinequery(G)$;
3 $PB = getboundary(S)$;
```
/* delete the original skyline to generate G' */
```
4 $G' = G$ - S;
5 $I = skylinequery(G')$;
6 $PB' = getboundary(I)$;
7 $C = getcandidateset(PB', I)$;
8 **for** each $candidate \in C$ **do**
9 $\quad$ $cost = Promote(candidate)$;
10 $\quad$ $minCost = min(minCost, cost)$;
11 **return** optimal members and corresponding promotion plans

---

Proof. Let n be such a member that belongs to neither I_G nor the set $\{c|\ c \notin S_G \wedge m(c) \in B_{I_G}\}$. Then c strictly dominates n, namely, $d_{in}(c) > d_{in}(n)$ and $d_{out}(c) > d_{out}(n)$. Let p is the optimal promotion plan which can promote c to be c', which belongs to the skyline, and $d_1 = d_{in}(c') - d_{in}(c)$, $d_2 = d_{out}(c') - d_{out}(c)$ as the count of incoming edges and outgoing edges respectively. Thereby $\zeta(c) = d_1 + d_2$. $\zeta(n) \geq d_1' + d_2'$, here $d_1' = d_{in}(c') - d_{in}(n) < d_1$, $d_2' = d_{out}(c') - d_{out}(n) < d_2$. Apparently, $\zeta(n) < \zeta(c)$.

We can employ Algorithm 1 to perform the promotion process against each candidate and then obtain the optimal candidate and corresponding promotion plans eventually by Algorithm 2.

4.2   Time Complexity Analysis

Theorem 2. *The worst time complexity of Algorithm 1 is $O(|S_G|)$.*

Proof. It takes $O(|S_G|)$ time to calculate the intersections and search the projective points which belong to the intercepted part of B_{S_G} in the worst case. Actually, we employed the binary search approach to improve it to $O(\log |S_G|)$. And searching the optimal candidate also takes $O(|S_G|)$ time. If new dominators of the candidate are generated so that we have to try adding more edges, which is an extremely rare case, we only need to add a limited number of edges to get the candidate successfully promoted because adding one edge will increase different dimensions of the two nodes respectively, and thus the dominance relationships are bound to break. Therefore, the worst time complexity of Algorithm 1 is $O(|S_G|)$.

Theorem 3. *The worst time complexity of Algorithm 2 is $O(|V|^2)$.*

Proof. Among the common algorithms for skyline query, the worst time complexity is $O(|V|^2)$. And it takes $O(|S_G|)$ time to calculate the promotion boundary, $O(|V| - |S_G|)$ time to determine the candidate set whose size is $|C_G|$. Consequently, the time complexity of Algorithm 2 is $O(|V|^2) + O(|S_G|) + O(|V| - |S_G|) + |C_G| \bullet O(|S_G|)$, namely the worst time complexity is $O(|V|^2)$.

It is apparent that the efficiency of the InfraSky algorithm is improved by orders of magnitude compared to the brute-force algorithm.

5   Experiments

In this section, we will present 2 different experiments which make comparisons on time cost and promotion cost respectively to show the superiority of the InfraSky algorithm on both real and synthetic datasets.

5.1   Experimental Setup

All the experiments were implemented using Java with the jdk version 1.6.0_10 and conducted on an Intel (R) Core(TM)2 Duo CPU T7300 @ 2.0GHz machine with 1 Gbytes Ram and 120 Gbytes Hard disk running Windows XP.

Datasets: 1) *wiki-Vote*: The dataset[1] contains the administrator elections and vote history data in *wikipedia* community. It consists of 2,794 elections with 103,663 total votes and 7,066 users participating in the elections (either casting a vote or being voted on). Nodes in the network represent *wikipedia* users and a directed edge from node i to node j represents that user i voted on user j.

2) *Power-law Set*: We used the graph data generator gengraph_win[2] to generate graph datasets in which the degrees of the nodes obey power law distribution. More specifically, gengraph_win generates a degree sequence which is composed by a set of integers according to several parameters assigned by user. The parameters include the count of degree numbers n, the minimum degree min, the maximum degree max and exponent of the power law distribution α. In the experiments, we generated a series of graph datasets whose degree distributions all obey power law distribution in order to analog social networks in different size respectively with different parameter combinations. As to each scale, after carrying out a large number of experiments by tuning the parameters respectively, we decide to set max to a random number between $30\% \times n$ and $50\% \times n$, and $alpha$ to a random number in [1, 2.5] to construct a representative network dataset on the basis of both the structure of the network and the efficiency of processing the dataset.

5.2   Experimental Results

Evaluation on Time Cost. We conducted this experiment to compare the time costs of the brute-force algorithm, the best so far IDP algorithm and the

---

[1] http://snap.stanford.edu/data/wiki-Vote.html
[2] http://www.cs.sunysb.edu/~algorith/implement/viger/distrib/

(a) smaller datasets

(b) larger datasets
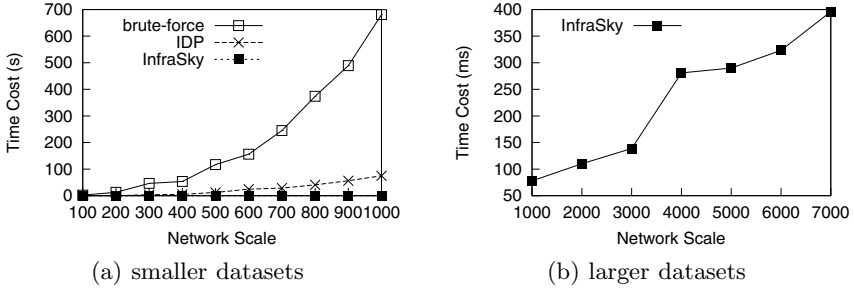
Fig. 4. Time cost comparison on wiki-Vote

(a) smaller datasets

(b) larger datasets
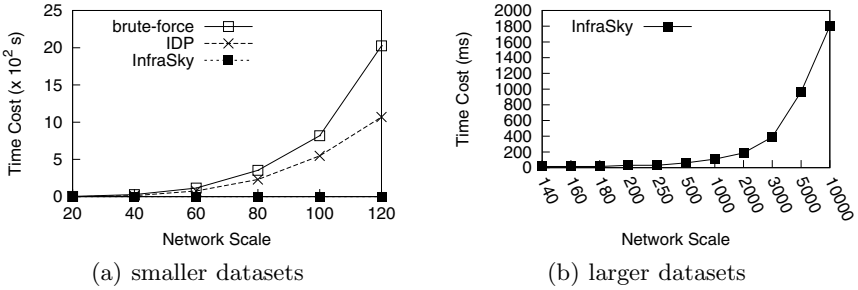
Fig. 5. Time cost comparison on Power-law Set

InfraSky algorithm. Figure 4 and Figure 5 show the differences of time costs between the algorithms in *wiki-Vote* and *Power-law Set* respectively. All the time costs obtained in this experiment are the average value of 10 different networks of the same scale, which are generated by tuning parameters.

Figure 4(a) and Figure 5(a) show that the InfraSky algorithm remarkably defeats the other two algorithms in both datasets. The larger the scale of the network is, the more obvious the superiority in efficiency of the InfraSky algorithm is. Furthermore, the brute-force algorithm and the IDP algorithm is limited by the scale of the network. Figure 4(b) and Figure 5(b) present that the time costs of brute-force algorithm and IDP algorithm become intolerable as the scale increases to some extent in both datasets so that the time can not even be displayed in the figures. In the mean time, the InfraSky algorithm still can finish within an acceptable time, which further proves its high efficiency and universality. As shown in Figure 4(b), for instance, when the scale of the network reaches 2000, the time cost of the brute-force algorithm is too high to tolerate, actually it takes far more than 2 hours. And the IDP algorithm also needs more than 10 minutes while the InfraSky algorithm can finish in 78 milliseconds. So the results of both two algorithm are even not able to be shown in the figure. As the scale grows to 7000, the time cost of our algorithm can still finish in less than 400 milliseconds. The result is similar in the Power-law Set as presented in Figure 5(b). Obviously, the efficiency of the promotion process is greatly

(a) wiki-Vote   (b) Power-law Set
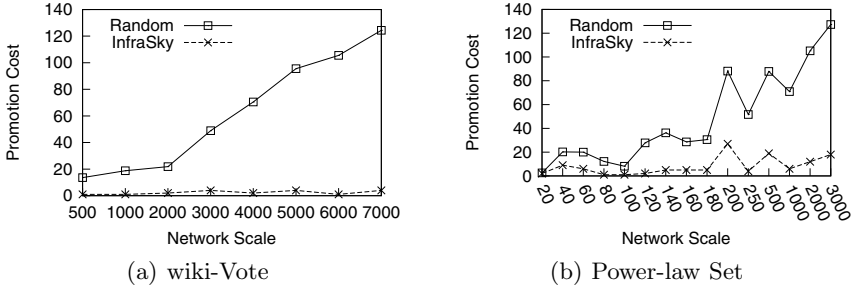
Fig. 6. Promotion cost comparison

improved by using the InfraSky algorithm. And the overall trend is in accord
with the time complexity of the InfraSky algorithm.

Evaluation on Promotion Cost. In this experiment we first randomly picked
one of the non-skyline members, and then we kept adding edges one by one, which
were randomly selected from the available edges corresponding to the candidate,
before the candidate was successfully promoted. The average count of added
edges in 10 repeated experiments was taken as the promotion cost result of the
method. Then, we ran the InfraSky algorithm to get the minimum promotion
cost. Figure 6(a) and Figure 6(b) illustrate the comparison of the promotion
costs on both datasets respectively.

As shown in Figure 6(a) and Figure 6(b), it takes higher costs to promote the
candidate successfully with the random promotion method against random non-
skyline candidate than the InfraSky algorithm which always provides the optimal
promotion cost. Even in a small-scaled network which contains only 500 nodes,
the promotion cost of the InfraSky algorithm is only 1, while it takes 13.6 edges
in average to finish the promotion in the random way on *wiki-Vote*, and 2 for
the InfraSky algorithm, 2.5 for the random method on a synthetic network with
20 nodes in *Power-law Set*. As the scale of the network increases, the difference
between the two promotion costs grows rapidly in general. In addition, because
of the different parameters and the random network data, the overall trends of
the promotion costs generated by the two methods both show a certain degree
of volatility in Figure 6(b). However, the promotion costs of the two methods
vary in the same pattern as the scale changes, which means both methods are
able to reflect the changes of the network correctly.

6   Conclusions

According to the characteristics of the skyline, we proposed several new concepts
such as infra-skyline and promotion boundary. Based on that, we proposed an
effective approach, namely the InfraSky algorithm, to solve the problem of mem-
ber promotion in SNs. The theoretical analysis and extensive experiments show
the effectiveness and efficiency of our InfraSky algorithm.

The direction of the future study above all is to provide an effective promotion algorithm in the social networks in which the weights of the edges are not equal. Secondly, we plan to introduce new metrics into the skyline operator which may bring high complexity to the problem. Besides, it will be an interesting problem of great applicable value when it allows more than one member to promote concurrently.

References

1. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE, pp. 421–430 (2001)
2. Chen, L., Lian, X.: Dynamic skyline queries in metric spaces. In: EDBT, pp. 333–343 (2008)
3. Deng, K., Zhou, X., Shen, H.T.: Multi-source skyline query processing in road networks. In: ICDE, pp. 796–805 (2007)
4. Fuhry, D., Jin, R., Zhang, D.: Efficient skyline computation in metric space. In: EDBT, pp. 1042–1051 (2009)
5. Jang, S., Park, C., Yoo, J.: Skyline Minimum Vector. In: APWeb, pp. 358–360 (2010)
6. Jin, W., Han, J., Ester, M.: Mining Thick Skylines over Large Databases. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 255–266. Springer, Heidelberg (2004)
7. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: VLDB, pp. 275–286 (2002)
8. Kung, H., Luccio, F., Preparata, F.: On finding the maxima of a set of vectors. Journal of the ACM (JACM) 22(4), 469–476 (1975)
9. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: SIGMOD, pp. 467–478 (2003)
10. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM TODS 30(1), 41–82 (2005)
11. Peng, Z., Wang, C., Han, L., Hao, J., Bai, Y.: Discovering the Most Potential Stars in Social Networks. In: Proceedings of the Third International Conference on Emerging Databases, Incheon, Korea (August 2011)
12. Sharifzadeh, M., Shahabi, C.: The spatial skyline queries. In: VLDB, pp. 751–762 (2006)
13. Tan, K.-L., Eng, P.-K., Ooi, B.C.: Efficient progressive skyline computation. In: VLDB, pp. 301–310 (2001)
14. Wu, T., Xin, D., Mei, Q., Han, J.: Promotion analysis in multi-dimensional space. PVLDB 2(1), 109–120 (2009)
15. Zou, L., Chen, L., Özsu, M.T., Zhao, D.: Dynamic Skyline Queries in Large Graphs. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5982, pp. 62–78. Springer, Heidelberg (2010)

Feature Based Informative Model for Discriminating Favorite Items from Unrated Ones

Bing Cheng, Tianqi Chen, Diyi Yang, Weinan Zhang,
Yongqiang Wang, and Yong Yu

Computer Science and Engineering,
Shanghai Jiao Tong University,
Dongchuan Road, Minhang District, Shanghai, China
{chengb,tqchen,yangdiyi,wnzhang,wangyq,yyu}@apex.sjtu.edu.cn

Abstract. In this paper, we describe a feature based informative model to the second track of this year's KDD Cup Challenge[1]. The goal is to discriminate songs rated highly by the user from ones never rated by him/her. The informative model is used to incorporate different kinds of information, such as taxonomy of items, item neighborhoods, user specific features and implicit feedback, into a single model. Additionally, we also adopt ranking oriented SVD and negative sampling to improve prediction accuracy. Our final model achieves an error rate of 3.10% on the test set with a single predictor, which is the best result of single predictors in all the publicized results on this task, even better than many ensemble models.

Keywords: Feature Based Informative Model, Ranking oriented SVD, Negative Sampling.

1 Introduction

The explosive growth of available choices from content providers has given great prominence to recommendation systems. In the past years, recommendation systems have shown great potential to help users find interesting items from large item space[1,2]. Due to its great benefits to both users and content providers, recommendation systems have been actively researched since it was introduced[3,4].

For this year's KDD Cup Challenge, Yahoo! Labs released a large music rating dataset. The contest consists of two tracks. The first track is a rating prediction problem that aims at minimizing RMSE (Root Mean Square Error). It is similar to the famous Netflix Prize Challenge[2]. The task of the second one is to discriminate the 3 songs rated highly by the user from the 3 ones never rated by her. In this task "rate highly" means a rating greater than or equal to 80. We tackle this problem as a top-n recommendation problem. That is, the three songs with higher prediction scores are regarded as the user's favorite songs, while the other 3 songs are considered to be unrated.

---

[1] http://kddcup.yahoo.com/
[2] www.netflixprize.com

In this paper, we use ranking oriented SVD to solve this problem. A negative sampling technique is utilized to further improve prediction accuracy. Most importantly, we propose to use a feature based informative model to incorporate different kinds of information into a single model. The ensemble of many algorithms is a useful approach to improve the overall performance. This has been proved by the winners of the Netflix Prize[5]. Different algorithms capture different information of the dataset, so they blend well. All the publicized results on KDD Cup Track2 also adopt ensemble techniques to boost their final predictions. However, ensemble usually needs extra computation cost. There is no doubt that a single model with comparable performance to ensemble models is more acceptable. Here we propose such a model. Different kinds of information, such as taxonomy of items (more on this in Section 2.2), item neighborhoods, user specific features and implicit feedback, are integrated into a single model. With this model, we achieve an error rate of 3.10% on the test set. This is the best result of single predictors among all publicized results on this task, even better than the performance of many ensemble models.

The reminder of this paper is organized as follows. In Section 2 we present the preliminaries, stating our task and giving some key properties of the dataset. Symbols that will be used later are also defined in this section. Ranking oriented SVD and the negative sampling strategies are detailed in Section 3. Section 4 focuses on the feature based informative model. In Section 5 we give our experimental results. Related works are summarized in Section 6. Finally in Section 7 we conclude our work.

2   Preliminaries

2.1   Problem Statement

For each user in the test set, six songs are given. Three of them are selected randomly from the songs rated highly by the user, and the other three are selected with probability proportional to the number of high ratings the song receives. The motivation for this setting is, for popular songs, user may have already heard about them. If he/she still doesn't rate these songs, it is likely that he/she doesn't like them. Participants need to separate the three positive examples from the three negative ones. The evaluation metric is error rate. That is, the ratio of wrongly classified test cases with respect to the total number of test cases.

2.2   Key Properties of the Yahoo! Music Dataset

For the task of binary user preference prediction, Yahoo! Labs released a dataset consisting of 67 million ratings from 24 thousand users on 30 thousand items. An important property of this dataset is the taxonomy information. That is, items have four categories: artist, album, song and genre. As we will show in Section 5, the taxonomy information can decrease error rate significantly.

2.3   Notation Definition

We consider the whole rating dataset $\mathbf{R} = [r_{u,i}]$ to be a sparse matrix. The letter u and i are used to denote user and item respectively. $r_{u,i}$ is user $u's$ rating on item i, $0 \leq r_{u,i} \leq 100$. Bold letters are used for matrices and vectors, non bold for scalars. The inner product of two vectors $\mathbf{p}$ and $\mathbf{q}$ is $< \mathbf{p}, \mathbf{q} >$. The letter $\mathbb{U}$ is used to represent the whole user set and the letter $\mathbb{I}$ for the whole item set. Predicted rating for (u,i) pair is $\hat{r}_{u,i}$. The set of users who rated item i is $\mathbb{U}(i)$ and the set of items rated by user u is $\mathbb{I}(u)$. $N(i;k)$ is the set of top-k neighborhoods of item i computed by Pearson Correlation Coefficient.

3   Prediction Models

In this section we will elaborate the ranking oriented SVD model with negative sampling for the binary user reference prediction problem. In Section 3.1 we briefly present classical SVD models that try to minimize RMSE. Section 3.2 focuses on the ranking oriented SVD models and the strategies to pair rating records. Finally in Section 3.3 we introduce the negative sampling approach used in this paper.

3.1   Classical *SVD* Models

Classical SVD [6] models mainly focus on approximating the rating of user u on item i by

$$\hat{r}_{u,i} = \mu + b_u + b_i + < \mathbf{p}_u, \mathbf{q}_i > \tag{1}$$

Here μ is the global average of the rating dataset $\mathbf{R}$, b_u and b_i are user and item bias respectively. The two vectors $\mathbf{p}_u, \mathbf{q}_i \in \mathbb{R}^f$ are f dimensional vectors used to capture the latent attributes of users and items. The parameters are learnt by minimizing RMSE.

$$L = \sum_{u,i} (r_{u,i} - \hat{r}_{u,i})^2 + regularization\ terms \tag{2}$$

We denote SVD models that try to minimize loss function (2) by *ReSVD* (Regression SVD) .These models gained great success in the Netflix Prize. However, as shown by [7], they usually don't work as well in the choice prediction problem.

3.2   Ranking Oriented SVD

A natural solution to the choice prediction problem is learning to rank. Eigen-Rank [8] and pLPA (probabilistic Latent Preference Analysis) [9] are two such models. Ranking oriented SVD models are first proposed by Nathan N. Liu et al.[10]. The key technique in these models is to turn either implicit or explicit feedback into user dependent pairwise preference regarding items. The pairwise preference is denoted by the variable δ_{uij}, which takes value of +1 if user u

prefers item i to item j, and -1 if the opposite holds. For our task of binary user preference prediction, we derive the pairwise preference by two ways. The first is based on the gap between two ratings, and the second uses two boundaries. These two ways correspond to the definition of δ_{uij} in Equation (3) and (4). We denote them as GAP_PAIR and $BOUND\_PAIR$ respectively.

$$\delta_{uij} = \begin{cases} +1 & i, j \in \mathbb{I}(u) \ and \ r_{u,i} - r_{u,j} \geq t \\ -1 & i, j \in \mathbb{I}(u) \ and \ r_{u,j} - r_{u,i} \geq t \end{cases} \tag{3}$$

$$\delta_{uij} = \begin{cases} +1 & i, j \in \mathbb{I}(u) \ and \ r_{u,i} \geq t_{lb} \ and \ r_{u,j} \leq t_{ub} \\ -1 & i, j \in \mathbb{I}(u) \ and \ r_{u,j} \geq t_{lb} \ and \ r_{u,i} \leq t_{ub} \end{cases} \tag{4}$$

Here t, t_{ub} and t_{lb} are pre-defined thresholds. In Section 5 we give our experimental results on these two definitions of δ_{uij}. Given a certain definition for δ_{uij}, we denote the set of triples (u, i, j) with $\delta_{uij} = +1$ as the set $\mathbb{D}$. $\mathbb{D}$ is used as the input for our ranking oriented SVD models.

We follow the work of [10] to use Bradley-Terry model [11] to design the loss function for the preference prediction problem. Under this model, each user u is associated with $|\mathbb{I}|$ parameters $\gamma_{u,i}$ indicating the utilities of these items to the user. The higher $\gamma_{u,i}$ is compared to $\gamma_{u,j}$, the more likely that $\delta_{uij} = +1$. This relation can be described using a sigmoid function:

$$P(\delta_{uij} = +1) = \frac{1}{1 + e^{-(\gamma_{u,i} - \gamma_{u,j})}} \tag{5}$$

To adopt the this model to SVD, we take $\gamma_{u,i} = \hat{r}_{u,i}$. This parametrization leads to the following maximum likelihood training procedure:

$$L = \sum_{(u,i,j) \in \mathbb{D}} ln(1 + e^{-(\hat{r}_{u,i} - \hat{r}_{u,j})}) + regularization \ terms \tag{6}$$

We denote SVD models that optimize the loss function defined in (6) as $RaSVD$ (Ranking oriented SVD).

3.3   Negative Sampling for SVD Models

Negative sampling for CF is proposed by Rong Pan et al. [12] to solve the OCCF problem. In OCCF, only positive examples are available. Such situations include users' visiting history to news page recommendation, users' clicking and trading history on online shopping site.

The negative sampling strategy we employ is user-oriented pairwise sampling which proves to work well in [12]. For every (u, i) pair in $\mathbf{R}$ with a rating higher than a pre-define threshold θ, we select k items not rated by u as the negative examples for this user. The negative examples are selected using the same approach as the negative examples in the test data. That is, the probability of an item being selected as negative example is proportional to the number of high ratings it receives in the rating set $\mathbf{R}$. The score of negative example is 0. Such positive/negative item pairs are used as the training data for our ranking oriented SVD models. We will show the impact of k and θ on prediction accuracy in Section 5.

4   Feature Based Informative Model

In this section we elaborate the informative model we use in this paper. Section 4.1 presents a feature based collaborative filtering framework (abbreviate to *FCFF*)[13], which serves as the basis of our informative model. Following sections will focus on how to incorporate different kinds of information into the framework.

4.1   Feature Based Collaborative Filtering Framework

Our informative model is based on the feature based collaborative filtering framework proposed by Chen at al. [13]. The input format of *FCFF* is similar to LibSVM[14], users just need to define features to implement new model. The model equation of *FCFF* is:

$$y(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \mu + \left(\sum_j b_j^g \alpha_j + \sum_j b_j^u \beta_j + \sum_j b_j^i \gamma_j \right) + \left(\sum_j \beta_j \mathbf{p}_j \right)^T * \left(\sum_j \gamma_j \mathbf{q}_j \right) \tag{7}$$

We can see from Equation (7) that in *FCFF*, the features can be divided into three groups: $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$. They represent global, user and item dependent features respectively. The corresponding $\mathbf{b}^g, \mathbf{b}^u, \mathbf{b}^i$ are the weights of these features, which need to be learnt by minimizing the loss function (2) or (6). Another important property of *FCFF* is that global features are not involved in factorization. They are linear features. Incorporating these features into SVD models can help us capture information from different sources. We will show this in the following sections.

For the basic SVD model defined in Equation (1), the features for *RaSVD* are defined in Equation (8):

$$\beta_h = \begin{cases} +1 & h = index(userId) \\ 0 & otherwise \end{cases} \qquad \gamma_h = \begin{cases} +1 & h = index(posId) \\ -1 & h = index(negId) \\ 0 & otherwise \end{cases} \tag{8}$$

Here *index* is a function used to map feature to a unique ID. In Equation (8) *posId* is the ID of the positive example and *negId* is the ID of the negative example. For *ReSVD* , features can be defined in a similar way. We omit the details due to space limitation.

4.2   Taxonomy-Aware *SVD* Models

As we mentioned in Section 2.2, an important property of the Yahoo! Music dataset is the taxonomy information. In particular, each song belongs to one album and one artist. Each album belongs to one artist. Moreover, every song or album has zero or multi genres. Intuitively, items that are closely correlated in the taxonomy may receive similar ratings from the same user. For example, given

that a user rates highly on an artist(album), we can conclude that it's likely that the user will also rate highly on the songs belonging to the artist(album). In this section we integrate this information into *FCFF*.

To capture the taxonomy relationship, for each item i in $\mathbb{I}$, we introduce a new bias term b_i' and latent vector $\mathbf{q}_i'$. The original bias term b_i and latent vector $\mathbf{q}_i$ are used to predict the score of the item itself, while the new bias term b_i' and latent vector $\mathbf{q}_i'$ are used to help the prediction of the children of item i. If i is artist, its children include the albums and songs of artist i. If i is album, its children are all the songs belonging to album i. This leads to the following formulation of the estimation of $r_{u,i}$:

$$\hat{r}_{u,i} = \mu + b_u + b_i + b_{Al(i)}' + b_{Ar(i)}' + < \mathbf{p}_u, \ \mathbf{q}_i + \mathbf{q}_{Al(i)}' + \mathbf{q}_{Ar(i)}' > \qquad (9)$$

Here $Al(i)$ is the album of item i, and $Ar(i)$ is the artist of item i. For rank models, the features of this taxonomy-aware SVD can be defined as follows:

$$\gamma_h = \begin{cases} +1 & h = index(posId) \ or \ h = index(Al(posId)) \ or \\ & h = index(Ar(posId)) \\ -1 & h = index(negId) \ or \ h = index(Al(negId)) \ or \\ & h = index(Ar(negId)) \\ 0 & otherwise \end{cases} \qquad (10)$$

$\boldsymbol{\beta}$ is defined the same way as Equation (8).

4.3   Integrating Implicit Feedback

As pointed out by early works [15,3], implicit feedback has great potential to improve the performance of recommendation systems. Compared to explicit feedback where explicit ratings are required, implicit feedback emphasizes more on which items the user rates. After integrating implicit feedback, prediction score for unknown (u, i) pair is:

$$\begin{aligned} \hat{r}_{u,i} = \mu + b_u + b_i + b_{i,Al(i)}' + b_{i,Ar(i)}' + < \mathbf{p}_u + \frac{1}{\sqrt{|\mathbb{I}(u)|}} \sum_{j \in \mathbb{I}(u)} \mathbf{q}_j'', \\ \mathbf{q}_i + \mathbf{q}_{Al(i)}' + \mathbf{q}_{Ar(i)}' > \end{aligned} \qquad (11)$$

In this model, each item i is associated with a new latent vector $\mathbf{q}_i''$, which is used to uncover information embedded in users' implicit feedback.

4.4   Integrating Neighborhood Information

In this section we will show how to integrate neighborhood information into the feature based informative model. Combining neighborhood and SVD model into a single predictor is first proposed by Y. Koren et al. [3] with the following model:

$$\begin{aligned} \hat{r}_{u,i} = \mu + b_i + b_u + < \mathbf{p}_u, \mathbf{q}_i > + \frac{1}{\sqrt{|\mathbb{R}(u,i;k)|}} \sum_{j \in \mathbb{R}(u,i;k)} (r_{u,j} - \hat{b}_{u,j}) w_{i,j} + \\ \frac{1}{\sqrt{|\mathbb{N}(u,i;k)|}} \sum_{j \in \mathbb{N}(u,i;k)} c_{i,j} \end{aligned} \qquad (12)$$

Here $\hat{b}_{u,j}$ is a baseline estimator composed only by user bias and item bias. $w_{i,j}$ and $c_{i,j}$ are correlation coefficient between two items i and j. $\mathbb{R}(u,i;k) = \mathbb{N}(u,i;k) = \mathbb{I}(u) \bigcap \mathbb{N}(i;k)$.

For our binary user preference prediction problem, we find that implicit feedback is much more useful than explicit feedback. Integrating implicit neighborhood information into Equation (11) we get:

$$\hat{r}_{u,i} = \mu + b_u + b_i + b'_{Al(i)} + b'_{Ar(i)} + < \mathbf{p}_u + \frac{1}{\sqrt{|\mathbb{I}(u)|}} \sum_{j\in\mathbb{I}(u)} \mathbf{q}''_j,$$
$$\mathbf{q}_i + \mathbf{q}'_{Al(i)} + \mathbf{q}'_{Ar(i)} > + \frac{1}{\sqrt{|\mathbb{N}(u,i;k)|}} \sum_{j\in\mathbb{N}(u,i;k)} c_{ij} \tag{13}$$

To implement the model described in (13) under *FCFF*, we just need to add new global features.

$$\alpha_h = \begin{cases} \frac{1}{\sqrt{\mathbb{N}(u,posId;k)}} & g \in \mathbb{N}(u, posId; k) \ and \ h = index(posId, g) \\ -\frac{1}{\sqrt{\mathbb{N}(u,negId;k)}} & g \in \mathbb{N}(u, negId; k) \ and \ h = index(negId, g) \\ 0 & otherwise \end{cases} \tag{14}$$

Here $index(posId, g)/index(negId, g)$ is the feature ID of the neighborhood pair $(posId, g)/(negId, g)$.

4.5   Integrating Taxonomy Based Classifier

As we mentioned in Section 4.2, users' preferences on artist and album have great impact on their attitudes towards the corresponding songs. In this section, we integrate a taxonomy based classifier into our informative model. The classifier uses four user dependent features:

 – User's rating on the artist of the song. If not rated, this value is 0;
 – Whether the rating on the artist of the song is higher than 80;
 – User's rating on the album of the song. If not rated, this value is 0;
 – Whether the rating on the album of the song is higher than 80;

If we denote these four features by $b_{u,j}$ ($0 \le j \le 3$), the new model is:

$$\hat{r}_{u,i} = \mu + b_u + b_i + b'_{Al(i)} + b'_{Ar(i)} + < \mathbf{p}_u + \frac{1}{\sqrt{|\mathbb{I}(u)|}} \sum_{j\in\mathbb{I}(u)} \mathbf{q}''_j,$$
$$\mathbf{q}_i + \mathbf{q}'_{Al(i)} + \mathbf{q}'_{Ar(i)} > + \frac{1}{\sqrt{|\mathbb{N}(u,i;k)|}} \sum_{j\in\mathbb{N}(u,i;k)} c_{ij} + \sum_{j=0}^{3} b_{u,j} \tag{15}$$

To incorporate the taxonomy based classifier into *FCFF*, we just need to define new global features for $b_{u,j}$ in a similar way as Equation (14).

5   Experimental Results and Analysis

In this section we give our experimental results on the test set provided by Yahoo! Labs. In the test set, there are 101172 users. For each user, 6 songs are given. The total number of test cases is 607032. For each test case, we need to

label the song as either positive or negative example for the user. The evaluation metric is the ratio of wrongly labeled test cases with respect to the total number of test cases.

To encourage further research on the problem, we open source all the code and implementations of our experiments. The implementation of *FCFF* is available at our laboratory page[3]. The code used to generate features for all the models in Section 4 is also released[4].

5.1 Performance of Informative Model on Error Rate

In this section we show the effectiveness of informative model. Before giving the experimental results, we first introduce some abbreviations that will be used in later sections in Table 1.

Table 1. Explanation of abbreviations

(a) Loss function

ReSVD	Loss function (2)
RaSVD +GAP_PAIR	Loss function (6) δ_{uij} Equation (3)
RaSVD +BOUND_PAIR	Loss function (6) δ_{uij} Equation (4)

(b) Prediction models

BSVD	Basic SVD, Equation (1)
Tax-SVD	Taxonomy-Aware SVD, Equation (9)
+IMFB	Implicit feedback, Equation (11)
+Item-10NN	Top 10 item neighborhood, Equation (13)
+Tax-CLF	Taxonomy based classifier, Equation (15)

The "+" symbol in the table means "recursive combining". For example, the "+" in the last line of Table 1(b) means combining taxonomy-aware SVD, implicit feedback, item neighborhood and taxonomy based classifier into a single model. This is also our best prediction model. We denote this model as *InfoSVD*(Informative SVD).

Table 2 shows the error rate of different models with triple negative sampling. The results show that incorporating extra information into *FCFF*, such as taxonomy of items, item neighborhoods, user specific features and implicit feedback, can always help lower error rate. Taking *RaSVD+BOUND_PAIR* as an example, incorporating item taxonomy into *BSVD* decreases error rate by 10.8%. After integrating implicit feedback, a decrease of 7.5% in error rate is achieved. Item neighborhood and taxonomy based classifier bring in even larger improvement. By incorporating these two sources of information, error rate decreases by 25.1% and 20.3% respectively. In Table 3 we give the performance of

---

[3] http://apex.sjtu.edu.cn/apex_wiki/svdfeature
[4] http://apex.sjtu.edu.cn/apex_wiki/kddtrack2

Table 2. Prediction error rate(%) of informative model with triple negative sampling. Sampling threshold $\theta = 40$.

	ReSVD $f = 100$	RaSVD+GAP_PAIR $t = 20, f = 100$	RaSVD+BOUND_PAIR $t_{lb} = 20, t_{ub} = 0, f = 100$
BSVD	6.52	*6.42*	6.76
Tax-SVD	6.15	*6.03*	*6.03*
+IMFB	5.71	*5.52*	5.58
+Item-10NN	*3.98*	4.17	4.18
+Tax-CLF	3.76	3.40	*3.33*

Table 3. Prediction error rate(%) of informative model with no negative sampling

	ReSVD $f = 100$	RaSVD+GAP_PAIR $t = 20, f = 100$	RaSVD+BOUND_PAIR $t_{lb} = 20, t_{ub} = 0, f = 100$
BSVD	*23.99*	25.20	29.40
Tax-SVD	*20.79*	22.74	26.92
+IMFB	17.09	17.03	*16.51*
+Item-10NN	*14.23*	15.70	15.63
+Tax-CLF	*9.63*	12.19	13.78

the same models as Table 2 with no negative sampling. The results confirm our analysis above. To sum up, compared to *BSVD*, *InfoSVD* can at least decrease error rate by 42.3%.

5.2   Best Models

In this section we give the results of the best models for both regression SVD and ranking oriented SVD. The parameter settings are also presented. After this we give a comparison of our results and all the publicized results on this task.

For both regression SVD and ranking oriented SVD, the best prediction models are *InfoSVD* defined in Equation 15. This again confirms the effectiveness of *FBCF* in our user preference prediction problem. By integrating different sources of information into a single model, we can indeed lower error rate significantly.

Table 4. Best results for regression SVD and ranking oriented SVD

	Best Results	Parameters
ReSVD	**3.78**	$\theta = 60, k = 3, f = 100$
RaSVD+BOUND_PAIR	**3.16**	$\theta = 60, k = 5, t_{lb} = 20, t_{ub} = 0, f = 100$
RaSVD+GAP_PAIR	**3.10**	$\theta = 60, k = 5, t = 40, f = 300$

Table 4 shows the best results of *ReSVD* and *RaSVD*. θ is sampling threshold and k is the number of negative examples generated for each triple $(u, i, r_{u,i})$

Table 5. Comparison of error rate(%) between *InfoSVD* and all publicized results

Team name	Best single predictor	Ensemble model
InfoSVD	**3.10**	
National Taiwan University	4.04	2.47
The Art of Lemon	3.49	2.48
commendo	4.28	2.49
The Thought Gang	3.71	2.93
The Core Team		3.87
False Positives	5.70	3.89
Opera Solutions	4.45	4.38
MyMediaLite	6.04	4.49
KKT's Learning Machine	5.62	4.63
coaco		5.20

with $r_{u,i} \geq \theta$. An error rate of *3.10%* is achieved by *InfoSVD* with ranking oriented loss function defined in Equation 6.

Table 5 gives a comparison of the results of *InfoSVD* and the top 10 teams on the leaderboard. We can see from the table that *InfoSVD* achieves the lowest error rate among all the single predictors, outperforming the second one of 3.49% by 11.2%. Additionally, the performance of *InfoSVD* is even better than some ensemble models.

5.3   Impact of Negative Sampling and Ranking Oriented *SVD*

Comparing Table 2 and Table 3 we find that negative sampling can improve recommendation accuracy significantly. With triple negative sampling, the error rate of the best model decreases from 9.63% to 3.33%. Another interesting observation from Table 2 and Table 3 is the performance comparison of *ReSVD* and *RaSVD*. As we can see from these two tables, without negative sampling, *ReSVD* performs better in most cases. However, after bringing in extra negative examples, the performance of *RaSVD* grows faster and outperforms *ReSVD*.

Table 6 shows the effect of parameter k in negative sampling. We can see from Table 6 that increasing k can always bring in improvement in prediction accuracy, but the gain becomes less as k increases. Since the impact of θ on error rate is not so obvious when θ varies from 20 to 80, we omit the experiment results on different values of θ.

6   Related Work

CF algorithms fall into two categories: neighborhood model [16,3] and matrix factorization model [3,17]. For choice prediction problems, learning to rank [8,9,10] and negative sampling [12] have been proved to work well. EigenRank[8] and pLPA[9] are two typical learning to rank models. These models address the ranking problem directly without a rating prediction step. Other learning to

Table 6. Impact of k in negative sampling on error rate(%) with θ fixed to 60

	ReSVD $f = 100$	RaSVD+GAP_PAIR $t = 20, f = 100$	RaSVD+BOUND_PAIR $t_{lb} = 20, t_{ub} = 0, f = 100$
k=1	4.48	*4.09*	4.30
k=2	4.14	3.66	*3.65*
k=3	4.05	3.54	*3.40*
k=4	3.96	3.45	*3.22*
k=5	3.92	3.40	*3.16*

rank models[10,18,19] assign scores to items as traditional approaches, but the scores are just used to rank items, instead of approximating the real ratings. Rendel et al.[18] propose to use Bayesian probabilistic ranking model for top-n recommendation. Shi et al.[19] propose a list-wise learning to rank model with matrix factorization.

As we can see from experiments, our proposed feature-based informative model lowers error rate significantly compared to basic SVD. LibFM[20] also uses idea of feature-based factorization model. Compared to their model, our model distinguishes features types, which allows us to incorporate useful information such as neighborhood and taxonomy more naturally.

7   Conclusions and Future Work

In this paper, we mainly study the feature based CF framework for discriminating uses' favorite songs from ones unrated by her. Under this framework, new models can be implemented easily by defining features in the input data. With this framework, we achieve an error rate of 3.10% with a single predictor, which is the best performance of all single predictors on this task. The effectiveness of ranking oriented models and negative sampling are also presented.

For future work, we plan to investigate the performance of our informative model in a more general and practical scenario: top-n recommendation. In this case, metrics like recall and precision in information retrieval can be used.

References

1. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: WWW 2007: Proceedings of the 16th International Conference on World Wide Web, pp. 271–280. ACM, New York (2007)
2. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing 7, 76–80 (2003)
3. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2008, pp. 426–434. ACM, New York (2008)

4. Hofmann, T.: Latent semantic models for collaborative filtering. ACM Trans. Inf. Syst. 22, 89–115 (2004)
5. Andreas, T., Jahrer, M., Bell, R.M., Park, F.: The bigchaos solution to the netflix grand prize, pp. 1–52 (2009)
6. Funk, S.: Try this at home (2006),
   `http://sifter.org/~simon/journal/20061211.html`
7. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 39–46. ACM, New York (2010)
8. Liu, N.N., Yang, Q.: EigenRank: a ranking-oriented approach to collaborative filtering. In: SIGIR 2008: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, pp. 83–90 (2008)
9. Liu, N.N., Zhao, M., Yang, Q.: Probabilistic latent preference analysis for collaborative filtering. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, pp. 759–766. ACM (2009)
10. Liu, N.N., Cao, B., Zhao, M., Yang, Q.: Adapting neighborhood and matrix factorization models for context aware recommendation. In: Proceedings of the Workshop on Context-Aware Movie Recommendation, pp. 7–13. ACM, New York (2010)
11. Marden, J.I.: Analyzing and Modeling Rank Data. Chapman & Hall (1995)
12. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R., Scholz, M., Yang, Q.: One-class collaborative filtering. In: ICDM 2008 (2008)
13. Chen, T., Zheng, Z., Lu, Q., Zhang, W., Yu, Y.: Feature-based matrix factorization. Technical Report APEX-TR-2011-07-11, Apex Data & Knowledge Management Lab, Shanghai Jiao Tong University (July 2011)
14. Chang, C.-C., Lin, C.-J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011),
    `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
15. Oard, D., Kim, J.: Implicit feedback for recommender systems. In: Proceedings of the AAAI Workshop on Recommender Systems, pp. 81–83 (1998)
16. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, WWW 2001 (2001)
17. Paterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: 13th ACM Int. Conf. on Knowledge Discovery and Data Mining, Proc. KDD Cup Workshop at SIGKDD 2007, pp. 39–42 (2007)
18. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. Machine Learning (2009)
19. Shi, Y., Larson, M., Hanjalic, A.: List-wise learning to rank with matrix factorization for collaborative filtering. In: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys 2010 (2010)
20. Rendle, S.: Factorization machines. In: Proceedings of the 10th IEEE International Conference on Data Mining. IEEE Computer Society (2010)

Search for Minority Information from Wikipedia Based on Similarity of Majority Information

Yuki Hattori and Akiyo Nadamoto

Konan University
8–9–1 Okamoto, Higashinada-ku, Kobe, Hyogo, 658–8501, Japan
mn124005@center.konan-u.ac.jp, nadamoto@konan-u.ac.jp

Abstract. In this research, we propose a method of searching for minority information, which is less acknowledged and less popular, on the internet. We propose two methods to extract minority information. One is that of calculating relevance of content. The other is based on analogy expression. In this paper, we propose such a minority search system. At this time, we consider it necessary to search for minority information in which a user is interested. Using our proposed system, the user inputs a query which represents their interest in majority information. Then the system searches for minority information that is similar to the majority information provided. Consequently, users can obtain the new information that users do not know and can discover new knowledge and new interests.

1 Introduction

Much information exists on the internet: it is said that an information explosion is occurring. Obtaining information that has high awareness and a high profile can be accomplished easily on the internet using search engines. Nevertheless, it is difficult for us to obtain information which has less awareness and which has a low profile because there is little information about it. We have no clues to extract such information. However, we consider that some important information exists among that minority information. For example, sports can be regarded as including a few majority sports and many minority sports. Majority sports such as baseball, basketball, and football (soccer) are reported by media, and we know them well. Actually, however, they are only a few sports among many sports enjoyed throughout the world. Many minority sports exist throughout the world as well. Moreover, some information about them does exist on the web. They might be interesting sports for us, but we do not know how to know more about such minority sports. We consider it convenient for users and authors who transmit the minority information, when a search system can search for minority information from the web. In this paper, we propose such a minority search system. At this time, we consider it necessary to search for minority information in which a user is interested. Using our proposed system, the user inputs a query which represents their interest in majority information. Then the system searches for minority information that is similar to the majority information provided.

Consequently, users can obtain the new information that users do not know and can discover new knowledge and new interests. Our proposed method consists of 'relevance of content', 'analogy expression', and 'category filtering'. As described in this paper, we extract minority information from Wikipedia as a first step of the research because much information has been gathered in Wikipedia.

The flow of search minority information progresses as follows and as shown in Figure 1:

1. The user inputs majority information in which the user is interested.
2. The system extracts articles from Wikipedia that include analogy expressions of the user's input keywords. We designate this feature as analogy search.
3. It extracts Wikipedia articles related to the keyword. At this time, we calculate the relevance of content using a link graph and similarity degree. We designate this feature as relation search.
4. It regards the candidate of minority information articles which have few edits and few editors in the results of (2) and (3).
5. It calculates category filtering in the result of (4) and extracts minority articles. For our filtering, we use filtering of two types: LSP-method and Wikipedia category.

For example, if the user inputs 'football' as a majority information keyword, then the system seeks articles having a sentence of "....looks like football...." or similar content of articles about soccer in Wikipedia. It extracts as candidates for minority sports those articles which have few edits and few editors. Then, it performs sports filtering and extracts minority articles from the candidates. In this case, the results are 'Bandy', 'Goalball', and 'Cuju'.

The remainder of this paper is organized as follows: Section 2 presents discussion of related work; Section 3 explains methods of searching for minority information; Section 4 explains the prototype system, and section 5 explains experiments; we conclude this paper in Section 6.

2   Related Work

In this research, we propose a method of searching for minority information from Wikipedia. Ohshima et al.[1] proposed methods of searching for web pages that are regarded 'semantically' as 'siblings' with respect to given page examples. Their approach was designed to find pages that are similar in theme but which have different content from the given sample pages. They designated this as 'sibling page search'. We search for minority information that is similar to a user's input, but which has low acknowledgement and low popularity.

Some studies have calculated the importance degree and relevance degree of web pages using web link graphs. The most famous among the research efforts is page rank[2]. Wang et al.[3] proposed a content–link coupled clustering algorithm that clusters web pages by combining contents and link analysis. They studied the effects of out-links (from the web pages), in-links (to the web page), and terms on the final clustering results as well as how to combine these three parts
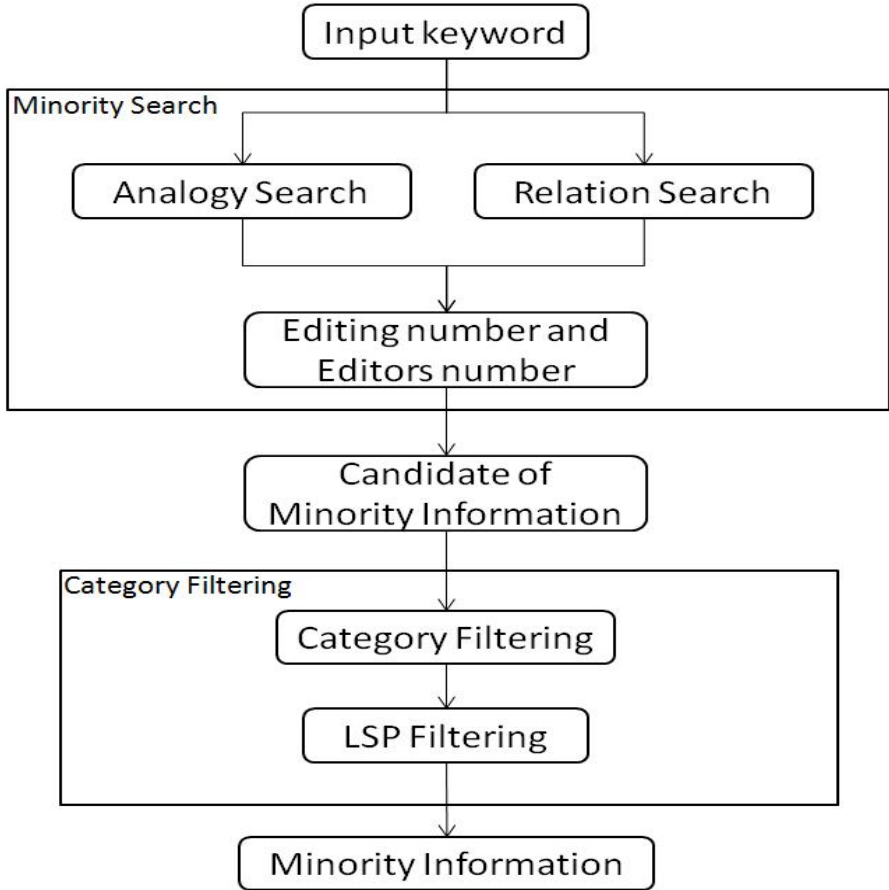
Fig. 1. System Flow

effectively and thereby improve the quality of clustering results. Glover et al.[4] analyzed the relative utility of document texts, and text in citing documents near a citation for purposes of classification and description.

Some studies have specifically examined the Wikipedia link graph. Milne et al.[5] proposed the Wikipedia link vector model (WLVM). It is unique in that it uses only the hyperlink structure of Wikipedia rather than its full textual content. Chernov et al.[6] proposed extraction of semantic information from Wikipedia by analyzing the links among categories. For this study, we use a link graph to extract related content. However our target research is extraction of minority information. It therefore differs from methods used in other studies.

Some studies have examined re-ranking. Zhuang et al.[7] proposed a Q-Rank to refine the ranking of search results effectively for any given query by constructing the query context from search query logs. Lee et al.[8] describe a model of an information retrieval system based on a document re-ranking method using

document clusters. Chidlovskii et al.[9] proposed a system for coupling a user and community profiling with the information search process. The search process and the ranking of relevant documents are accomplished within the context of a particular user or community perspective. The methods of their re-ranking methods resemble those used in our research. Nevertheless, the purpose of our research differs. We seek to extract minority information.

3   Minority Search

We first extract candidates of minority information that is similar to the user's input keyword from Wikipedia. At this time, we use methods of two types: 'relational search' and 'analogy search'.

3.1   Relational Search

The purpose of this study is searching for the minority content resembling the user's input majority content. We extract high relational content using a link graph and calculation of similarity. The following is the process used to extract the high relational content from Wikipedia.

1. We make articles of a query where the user inputs a basic node and creates a link graph of Wikipedia. At this time, each node represents an article.
2. We regard nodes which are linked to the basic node as related to the basic node. We designate the node as an inlink node. We regard nodes that are linked from the basic node as majority information; we therefore ignore those nodes.
3. We delete the inlink node, which has only one anchor text of basic node, because it is not closely related to the basic node.
4. We calculate the similarity between each inlink node and basic node using cosine similarity. If the similarity degree is greater than threshold α, then it becomes related information. We regard α as 0.35 by our experiment.

For example, as shown in Figure 2, we first delete nodes C and G because they are only linked from the basic node. Nodes D and F have only one anchor text of a basic node; they are deleted. The similarity degree of node H is less than 0.35. We delete it. In this case, nodes A, B, and E become related information.

3.2   Analogy Search

Sometimes minority information is extremely minority information that has only a small Wikipedia article or none at all. In this case, we can not extract it using relational search because the similarity is too small. However, as one characteristic of minority information, people sometimes explain it through comparison with majority information. For example, the article about "Sepak takraw" which

: The article that the user input（a standard node）
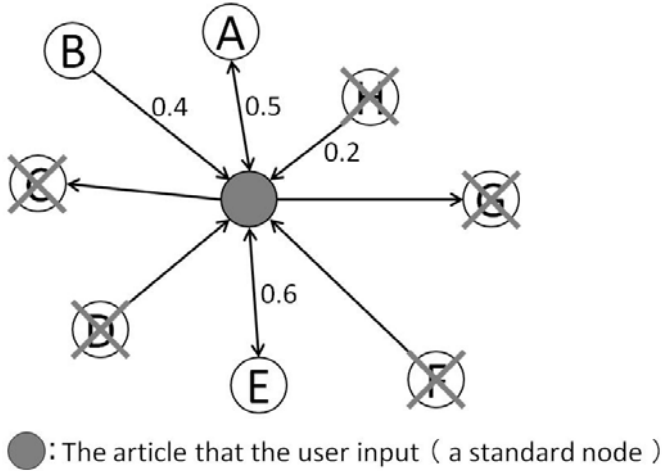
Fig. 2. Link Graph

is a kind of sport, is compared with 'volleyball'. The article of 'Goulash' which is Hungarian food is compared to 'stew'. We specifically examine the characteristics. Then we propose an analogy search. Comparison words are often used, such as "...like" and "similar to ...." Table 1 presents examples of comparison words.

Table 1. Examples of Comparison Words used in Wikipedia

Comparison Words	Example(article title)
...like	like football(Eaton field game)
similar ...of	similar sport of volleyball(Sepak takraw)
combines A and B	combines cross-country skiing and rifle shooting(Biathlon)
hybrid ...A with B	a hybrid sport that combines chess with boxing(Chess boxing)
...grew out of	grew out of skydiving(BASE jumping)

3.3   Extraction of Minority Information Candidates

We extract candidates of minority information from related information and anology information. The meaning of minority is "the smaller part or number; a number, part, or amount forming less than half of the whole.". As described in this paper, we regard less-acknowledged and less-popular information as minority information. However, we do not know what less-acknowledged and low popularity information are. When we extract minority information, we specifically examine the numbers of edits and editors of Wikipedia articles. Our hypothesis states that "Majority information is known by many peopled and the article of

it has many edits and many editors. However, minority information is known by few people and its related articles have few edits and few editors."

We conducted an experiment to prove that hypothesis. For this experiment, we use sports data because subjects can judge it easily. In our experiments, 13 subjects judged 150 sports that we selected randomly from Wikipedia. The subjects judged the results using the following three-step judgment process:

1: I know it.

0: I know only the name of it.

-1: I do not know it.

Table 2 presents some results of the experiment. The average is obtained by dividing the total of the results by the number of subjects. When the value of the average is near "1", many people know the sport. When the value of the average is near "-1", it means that few people know the sport. We regard the value of the average under "0" as the supervised data of minority sports.

Table 2. Experiment Results

No	Sports	Average	Evaluation:1	Evaluation:0	Evaluation:-1
1	Futsal	1	13	0	0
2	Handball	1	13	0	0
3	Water polo	0.8462	11	2	0
4	Polo	-0.6290	2	0	11
5	Rodeo	0.6154	8	5	0
6	Capoeira	0.2308	5	6	2
7	Fencing	1	13	0	0
8	Biathlon	-0.3850	3	2	8
9	Modern competitive archery	1	13	0	0
10	Darts	1	13	0	0
11	Lacrosse	0.8462	11	2	0
12	Squash	0.7692	10	3	0
13	Luge	-0.5380	2	2	9
14	Snowkiting	-0.9230	0	1	12
15	Canoe polo	-0.9230	0	1	12
:	:	:	:	:	:

After experimentation, we examine the numbers of edits and editors of each Wikipedia article. The results are shown in Table 3, Figure 3, and Figure 4. In Table 3, listed from No. 1 to No. 4 are majority sports. They have numerous edits and many editors. From No. 5 to number No. 8 are minority sports; they have few edits and few editors. In Figure 3 and Figure 4, the numbers of edits and editors increased, the sport whose result of experiment is '1' increases. However, where both the number of edits and editors decrease, sports for which the results of the experiment are '-1' increase. Results show that our hypothesis is correct, which means that the majority information has many edits and many editors and that minority information has few edits and few editors.

We measure the numbers of edits and editors of both the related information and analogy information. Subsequently, we extract candidates of minority information from them using thresholds. In our experiment, the threshold of the number of edits is 180; the threshold of the number of editors is 80.

Table 3. The Numbers of Edits and Editors

No.	Sports	Number of Edits	Number of Editors
1	Baseball	1087	507
2	Association football	880	411
3	Table tennis	874	389
4	Judo	555	267
5	Iaido	153	70
6	Petanque	95	62
7	Sepak takraw	76	46
8	Pesapallo	28	24
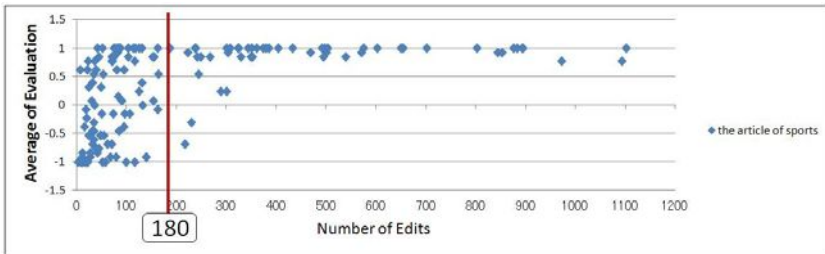
Fig. 3. Relation between the Number of Edits and the Average of the Evaluation

3.4 Category Filtering

Some different category articles exist among candidates of minority information. For example, when a user wants to know about minority sports and inputs majority sports, some minority sports team names, and equipment of the sports. They are not minority sports names. We must remove this information from candidates of minority information. We propose category filtering to extract minority information that a user wants to know. We first conduct filtering based on the Wikipedia category. Subsequently, we filter based on LSP methods.

Filtering Based on Category of Wikipedia
Wikipedia articles are divided by category. The category has a hierarchic structure. We specifically examine the category and extract two level above in the structure. First, we extract the category of the user's input query. Then we also extract a category of all candidate minority information. If a candidate of minority information differs in terms of the category of the query, then we remove it from minority information candidates.
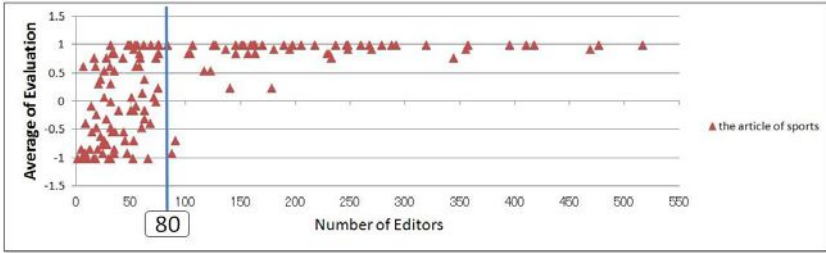
Fig. 4. Relations between the Number of Editors and the Average of the Evaluation

Filtering Based on LSP Method

Filtering based on category of Wikipedia can not remove all different minority information. For example, when the user inputs a sport name, equipment of sports sometimes can not be removed because the category of the equipment is 'sports'. Then we filter the information using LSP method, which was proposed by Nakayama et al.[10]. Nakayama et al. specifically examined the Wikipedia article grammar and proposed the LSP methods which regard the lead sentences of the article as important sentences. The lead sentences of almost all articles of Wikipedia consist of an is-a relation. The LSP method specifically examines the is-a relation and extracts upper concepts of the article. For example, an article of 'racquet' is "A racquet is a sports implement." The last words of 'implement' are the upper concept of 'racquet'. In this case, the upper concept is not a sports name. For that reason, we remove it from candidates of minority information. In this way, we use the LSP method to filter the different category candidates.

After filtering, the candidate of minority information becomes minority information.

4   Prototype System

We developed a prototype system using our proposed method. We used Ruby as the programming language, and used CGI as the user interface. Figure 5 shows the user input display and the output of the system. The first user input query and click search button. Then the system searches for the minority information that is similar to the query and displays the result. Results are displayed in the list and the user selects the article that the user wants to know.

5   Experiment

We conducted two experiments. One measured benefits of analogy search, the other measured benefits of our proposed system. In our experiments, we target sports because it is easy to distinguish majority sports from minority sports.
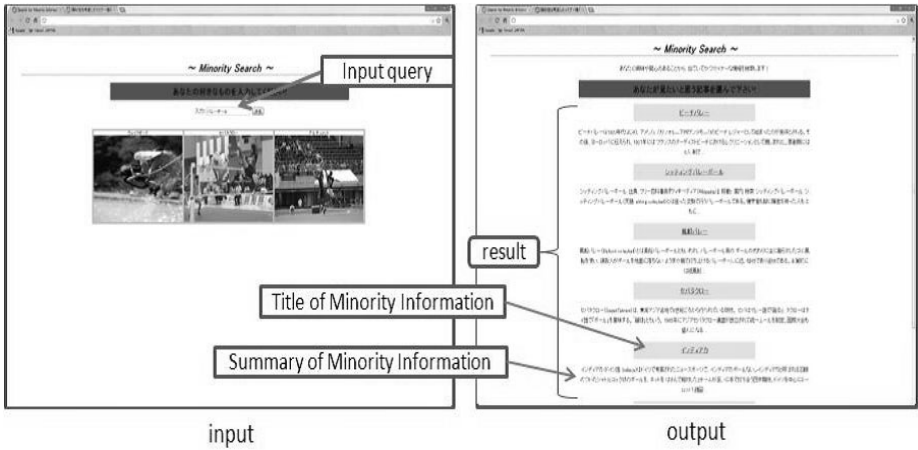
Fig. 5. Display of Prototype System

5.1   Experiment of Analogy Search

We used an experiment of analogy search to measure the system benefits. We measured the minority degree of the results of analogy search using the numbers of edits and editors. First, we input seven types of majority sports names and extracted analogy information using analogy search. Next we calculated the numbers of edits and editors.

Results and Discussion
Table 4 presents results of the experiment. Many articles extracted using analogy search were lower than the threshold of numbers of edits and editors described in section 3.3. Futsal and Beach volleyball are majority sports, but these sports are greater than the threshold numbers of edits and editors. Therefore, the system does not regard them as minority sports. In this way, our proposed analogy search and number of edits and editors is beneficial to extract minority information.

5.2   Experiment Conducted Using the Proposed System

We measured the benefit of our proposed system using the prototype system. We input majority sports of twelve types and calculated the precision, recall, and F-measure. This time, we regard 116 minority sports as the correct answer. The 116 minority sports comprise 59 minority sports that users judged in our experiment in section 3.3 and 57 minority sports obtained manually from Wikipedia.

Results and Discussion
Table 5 presents the results of our experiment. The average of the recall, precision, and F-measure show good results. However, the results of golf and football are not better than those of other sports because category filtering using LSP method is not good. Category filtering using LSP method examines only the first

Table 4. Results of analogy search

Input	Minority Sports that were acquired	Editing Number	Editor Number
Association football	Bandy	38	28
	Futsal	299	152
	Cycle ball	26	12
	Powerchair Football	33	21
	Freestyle football	5	3
Volleyball	Sepak takraw	77	47
	Peteca	43	25
	Paralympic volleyball	41	24
	Beach volleyball	135	80
Basketball	Streetball	44	24
	Beach basketball	6	5
	Water basketball	6	6
	Canoe polo	79	47
Baseball	Cricket	179	79
	Stickball	6	6
	Softball	354	94
Tennis	Tamburello	24	12
Rugby football	Rugby sevens	74	32
	Touch football	16	5
Judo	Sambo (martial art)	172	62
	Kurash	23	12

Table 5. Results of System Experiment

Sports	Recall	Precision	F-measure
Association football	50%	71%	59%
Volleyball	78%	88%	82%
Basketball	100%	100%	100%
Baseball	67%	100%	80%
Tennis	57%	100%	73%
Badminton	50%	100%	67%
Rugby football	17%	33%	22%
Golf	40%	100%	57%
Hockey	57%	75%	65%
Sumo	50%	75%	60%
Figure skating	100%	100%	100%
Modern competitive archery	38%	100%	55%
Average	59%	87%	68%

sentence. When the first sentence of an article has multiple verbs, the system can not ascertain the upper concept from it. Many articles are related to minority sports resembling golf or football have multiple verbs in the first sentence, the system therefore can not obtain the correct upper concept and delete them from minority information list.

On the other hands, one of the results of rugby is cricket. However, the cricket is not similar to the rugby. The reason why the cricket is included in the results is analogy search. There is a sentence which is "The manager can not instruct to players while playing a game like rugby" in the cricket article. In this case, 'like rugby' is not similar to the game. In this way, we have to consider the words of analogy search.

6   Conclusion

We proposed a method of searching for minority information that is less-acknowledged and has less popularity in Wikipedia. We first extracted related information that resembles the user's input query using related search and analogy search. Next we extracted minority information candidates using the numbers of edits and editors. Finally, we calculated category filtering and extracted minority information from Wikipedia. We developed a prototype system and experiments to measure the benefits of our proposed system.

In the near future we will study the following:

– Improvement of category filtering.
– Extraction of personalized minority information.
  The minority level differs among people. Therefore, we should consider personalized minority information.
– Extraction of minority information from the web.
  As described in this paper, we extracted minority information from Wikipedia as a first step of this research. However, the web includes large amounts of minority information and might not have related articles in Wikipedia. We should therefore devise some means to extract minority information from the web.

References

1. Ohshima, H., Oyama, S., Tanaka, K.: Sibling Page Search by Page Examples. In: Sugimoto, S., Hunter, J., Rauber, A., Morishima, A. (eds.) ICADL 2006. LNCS, vol. 4312, pp. 91–100. Springer, Heidelberg (2006)
2. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Proc. of the 7th International Conference on World Wide Web (WWW 1998), pp. 107–117 (1998)
3. Wang, Y., Kitsuregawa, M.: Evaluation Contents-Link Coupled Web Page Clustering for Web Search Results. In: Proceedings of the 7th International Conference on Information and Knowledge Management (2002)
4. Glover, E.J., Tsioutsiouliklis, K., et al.: Flake.: Using Web structure for classifying and describing Web pages. In: Proc. of WWW12 (2002)
5. Milne, D.: Computing Semantic Relatedness using Wikipedia Link Structure. In: Proceedings of the New Zealand Computer Science Research Student Conference, NZCSRSC 2007 (2007)

6. Chernov, S., Iofciu, T., Nejdl, W., Zhuo, X.: Extracting semantic relationships between wikipedia categories. In: 1st International Workshop: SemWiki 2006 - From Wiki to Semantics (SemWiki 2006), co-located with the ESWC 2006, Budva, Montenegro, June 12 (2006)
7. Zhuang, Z., Cucerzan, S.: Re-ranking search results using query logs. In: Proceedings of the 15th International Conference on Information and Knowledge Management (CIKM 2006), Arlington, Virginia, pp. 860–861 (2006)
8. Lee, K.S., Park, Y.C., Choi, K.S.: Re-ranking model based on document clusters. Information Processing and Management 37(1), 1–14 (2001)
9. Chidlovskii, B., Glance, N.S., Grasso, M.A.: Collaborative ReRanking of Search Results. In: AAAI-2000 Workshop on AI for Web Search (2000)
10. Nakayama, K., Hara, T., Nishio, S.: Wikipedia Mining - Wikipedia as a Corpus for Knowledge Extraction. In: Proceedings of Annual Wikipedia Conference, Wikimania (2008)

An Incremental Approach to Analyzing Temporal Constraints of Workflow Processes

Yanhua Du[1] and Xitong Li[2]

[1] School of Mechanical Engineering, University of Science and Technology Beijing,
Beijing 100083, China
[2] Sloan School of Management, Massachusetts Institute of Technology,
Cambridge, MA 02142 USA
duyanhua@ustb.edu.cn, xitongli@mit.edu

Abstract. The current fast-changing business environment requires workflow management systems to provide the ability of incremental analysis when process models are edited or updated. There are few researches on the dynamic verification of temporal constraints in an incremental way. In this paper, we present an approach to incremental analysis of temporal constraints. Firstly, after each operation performed by designers, the old change regions before the operation and the new change regions after the operation are generated by comparing the old and new Time Workflow nets (TWF-nets) of processes. Secondly, the sprouting graph recording time and path information of old TWF-net is maintained efficiently to the new state, instead of re-constructing a new sprouting graph. Finally, temporal violations are checked and the paths with violations are reported to designers. This approach is particularly applicable and efficient in terms of time and space for large-scale and complex processes.

Keywords: Temporal constraints, Time Workflow nets, Sprouting graphs, Incremental analysis.

1 Introduction

Recently, the fast-changing business environment requires workflow management systems (WfMSs) to have the ability to incrementally analyze workflow process models when they are being edited or updated, especially for large-scale and complex processes [1-2]. To assure the correctness of executing workflow processes, the analysis on structural errors [1-4] and temporal violations [5-9] are usually required. The verification of structural errors after changes are made has been recognized by workflow communities for a long time and different approaches have been developed [1-4]. In the literature, dynamic verification of temporal correctness after changes has not been fully investigated. The existing methods [5-9] can only answer whether there are temporal violations but they suffer from the following weaknesses:

(1) They assume that there is only one single workflow process and there are no resource dependencies. However, multiple workflow processes may execute concurrently under resource dependencies in a workflow management system in practice.

(2) The changes are limited to simple editing operations on activities. However, it is more often that several operations are changed simultaneously or a part of the workflow process is deleted or modified. None of them can deal with such cases.

(3) They are less efficient because they need to repeatedly investigate and calculate all of the processes after any change are made. In case of large-scale process models, they analyze temporal constraints inefficiently and cannot respond to users quickly.

In this paper, we present an incremental approach based on the sprouting graphs [9, 10] to check temporal violations in concurrent workflow processes. Firstly, the old change regions before the change and the new change regions after the change are generated by comparing the structures of the old and new TWF-nets. Here, change regions are the parts of the TWF-nets containing all the transitions directly or indirectly affected by the changes. Secondly, the sprouting graph is constructed and maintained efficiently with no need to calculate the whole new TWF-nets. It only needs to examine the change regions of TWF-nets and can check temporal violations by analyzing sprouting graph. Finally, temporal violations are checked based on the updated sprouting graph and the paths with violation are reported to designers.

This paper and the work in [9] are both based on the formalism foundation - sprouting graphs [9, 10], but they have different motivations and address different issues. The work in [9] focuses on dynamic checking temporal violations at run-time and provides the solution to change some transition time intervals. In contrast, this paper aims to deal with incremental analysis of temporal violations at build-time.

Compared with the existing methods [1-9], the main contributions of this paper are summarized as follows:

(1) We can dynamically check the temporal constraints of multiple workflow processes with resource dependencies, rather than a single workflow.
(2) We adopt the change regions of TWF-nets to deal with the complex situations of modifying TWF-nets, not only just simple editing operations.
(3) The sprouting graphs can be maintained efficiently with no need to calculate the whole new TWF-nets. Also, the results can be obtained by simply calculating.

2 Preliminary Formalism

2.1 TWF-Nets

In this paper, Petri net [2, 4, 9] is used to formalize workflow processes.

Definition 1. (WF-net): A *PN* is called a workflow net (WF-net) if and only if:
1) *PN* has two special places: ε and θ. Place ε is a source and $^{\bullet}\varepsilon = \varnothing$; θ is a sink and $\theta^{\bullet} = \varnothing$; and
2) If we add a new transition to *PN* which connects place θ with ε, i.e., $^{\bullet}t = \{\theta\}$, $t^{\bullet} = \{\varepsilon\}$, then the resulting *PN* is strongly connected.

Definition 2. (Time WF-net): A Time WF-net (TWF-net) is a 4-tuple (*P*, *T*, *F*, *FI*), where:
1) {*P*, *T*, *F*} is a WF-net;

2) *FI* is the set of pairs of nonnegative real numbers [*l*, *u*] related to each transition, which is used to represent the minimum firing time and the maximum firing time, respectively.

In the paper, TWF-net is based on the weak semantics of time PN, in which the firing of transitions can be freely chosen by decisions local to them, and independently from the conflicts with other ones. Assume transition $t \in T$ is enabled in a state M and is associated with a time interval [*l*, *u*], $(0 \le l \le u)$. Then, let s and $\tau(t)$ denote the enabled time and the actual firing time of t, respectively. We have $s+l \le \tau(t) \le s+u$.

In the real business fields, more than one workflow process usually exists in the WfMSs and there are resource dependencies among them. We assume that the FCFS (First Come First Served) policy is applied to allocate resources to activities [8, 9], i.e., an activity is allocated with the resource first if the transition representing that activity is enabled first. We use a resource place p_r to denote the shared resource and add corresponding directed arcs from/to the transitions to denote the request and release of the resource. $M_0(p_r)=1$, which means there is one resource instance at the initial marking. An example of TWF-net is shown in Fig.1, whose description is presented in Section 6.
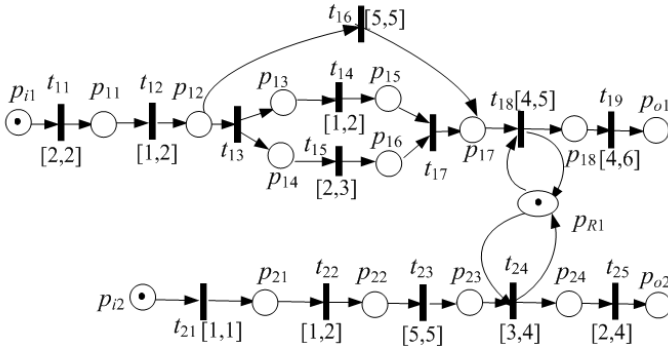
Fig. 1. An example of TWF-net

2.2   Temporal Constraints

Temporal constraints, e.g. assigned deadlines, usually are set explicitly by the process designer or enforced by law, regulations or business rules. Usually, the types of temporal constraints mainly include [6-9]: upper bound, lower bound and fixed-time.

For the sake of simplification, we use relative temporal constraints as the *canonical representation* for temporal constraints [6-9]. That is, we use $D_R(t_i, t_j) \le s$ (t_i is a reference point) to denote that t_j should end its execution no later than s time units after t_i starts.

Given a temporal constraint such as $D_R(t_i, t_j) \le s$, if the difference between enabling time interval of t_i and possible firing time interval of t_j is strictly less than or equals to s in all execution cases, it is *completely satisfied*. If it is strictly greater than s in all execution cases, then it is *completely violated*. Otherwise, $D_R(t_i, t_j) \le s$ is *partly satisfied* and has the possibility of being violated.

2.3   Sprouting Graphs of TWF-Nets

Definition 3. (Sprouting graph) [9,10]: A sprouting graph of a TWF-net W is defined as a 2-tuple (V, E), where V is a set of nodes and E is a set of directed arcs between nodes.

1) A node $v \in V$ is labeled (p, b) which corresponds to a place p in W. b is an ordered pair (Path_set, Time_set). Path_set is a set of paths. Time_set is a set of time intervals and each time interval in Time_set represents the time that the corresponding path in Path_set spends. If the ith path in the set of paths is $\varnothing$, then the ith time interval of the set of time intervals is $\varnothing$ as well.
2) A directed arc $e \in E$ is labeled (t, d) which corresponds to a transition t in W. And d is the time interval of t.

In a sprouting graph, an arc and its input/output nodes correspond to a transition and its input/output places. The set of time intervals of its input node is the possible enable time of the transition. The set of time intervals of its output node is the possible firing time of its corresponding transition. The set of paths in its input and output nodes include the paths of instances executing before and after its corresponding transition.

Sprouting graph not only has a more compact representation than a traditional timed state graph, but also contain both execution time and path of TWF-net [9,10]. So that we can check the temporal violations of workflow processes and find out the violation paths. Then, the designers can try to correct the execution of the activities in the path, e.g., to shorten the execution time of activities to solve violations [9].

Note that if TWF-net includes a loop structure, we transform it to a sequence of transition by approximating the number of loops in building sprouting graph.
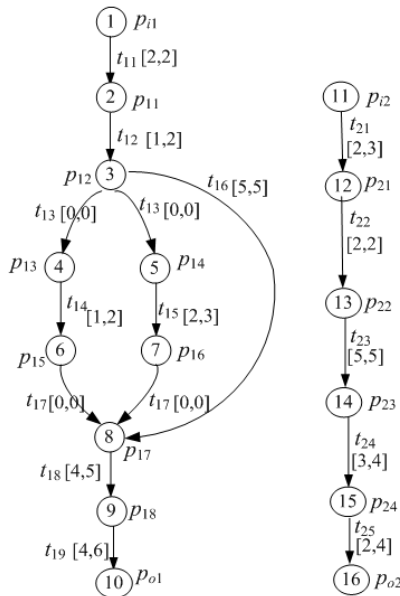
Fig. 2. The sprouting graph of TWF-net in Fig.1

For example, these two processes are specified as the Fig. 1. Based on [9], we construct the sprouting graph as shown in Fig. 2. The time and path sets of its nodes are shown in Table 1.

Table 1. The nodes of sprouting graph in Fig. 2

Node	Time sets	Path sets
1	$\varnothing$	$\varnothing$
2	$\{[2,2]\}$	$\{\{t_{11}\}\}$
3	$\{[3,4]\}$	$\{\{t_{11},t_{12}\}\}$
4	$\{[3,4]\}$	$\{\{t_{11},t_{12},t_{13}\}\}$
5	$\{[3,4]\}$	$\{\{t_{11},t_{12},t_{13}\}\}$
6	$\{[4,6]\}$	$\{\{t_{11},t_{12},t_{13},t_{14}\}\}$
7	$\{[5,7]\}$	$\{\{t_{11},t_{12},t_{13},t_{15}\}\}$
8	$\{[5,7],[8,9]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14}) \parallel (t_{11},t_{12},t_{13},t_{15}),t_{17}\}, \{t_{11},t_{12},t_{16}\}\}$
9	$\{[9,12],[12,14]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14}) \parallel (t_{11},t_{12},t_{13},t_{15}),t_{17}(t_{21},t_{22},t_{23}),t_{18}\},$ $\{t_{11},t_{12},t_{16}(t_{21},t_{22},t_{23}),t_{18}\}\}$
10	$\{[13,18],[16,20]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14}) \parallel (t_{11},t_{12},t_{13},t_{15}),t_{17}(t_{21},t_{22},t_{23}),t_{18},t_{19}\},$ $\{t_{11},t_{12},t_{16}(t_{21},t_{22},t_{23}),t_{18},t_{19}\}\}$
11	$\varnothing$	$\varnothing$
12	$\{[2,3]\}$	$\{\{t_{21}\}\}$
13	$\{[4,5]\}$	$\{\{t_{21},t_{22}\}\}$
14	$\{[9,10]\}$	$\{\{t_{21},t_{22},t_{23}\}\}$
15	$\{[12,16],[15,18]\}$	$\{\{t_{21},t_{22},t_{23}((t_{11},t_{12},t_{13},t_{14}) \parallel (t_{11},t_{12},t_{13},t_{15}),t_{17}),t_{24}\},$ $\{t_{21},t_{22},t_{23}(t_{11},t_{12},t_{16}),t_{24}\}\}$
16	$\{[14,20],[17,22]\}$	$\{\{t_{21},t_{22},t_{23}((t_{11},t_{12},t_{13},t_{14}) \parallel (t_{11},t_{12},t_{13},t_{15}),t_{17}),t_{24},t_{25}\},$ $\{t_{21},t_{22},t_{23}(t_{11},t_{12},t_{16}),t_{24},t_{25}\}\}$

Based on the sprouting graphs, we can check temporal constraints on TWF-nets. For $D_R(t_i, t_j) \leq s$, we need consider the time intervals from the input nodes of the arc corresponding to t_i and from output nodes of the arc corresponding to t_j. Here, we denote the time set of the input nodes of the arc corresponding to t_i is TSE_i, and the time set of the output nodes of the arc corresponding to t_j is TFS_j, respectively.

For $\forall [c,d] \in TSE_i$ and $\forall [a,b] \in TFS_j$, we obtain all the ordered pairs of time intervals of t_i enabling and t_j firing, such as $<[c,d], [a,b]>$. Then, for each ordered pair, we check if Y is before X where $X=s$, $Y=[a-d, b-c]$.

Case 1: $s<a-d$, the temporal constraint is completely violated under the current ordered pair.

Case 2: $b-c \leq s$, the temporal constraint is completely satisfied under the current ordered pair.

Case 3: $a-d \leq s < b-c$, the temporal constraint is partly satisfied and has the possibility of being violated under the current ordered pair.

Finally, if temporal constraint $D_R(t_i, t_j) \leq s$ is completely satisfied under all the ordered pairs, then it is *completely satisfied*. If it is completely violated under all the ordered

pairs, then it is *completely violated*. Otherwise, it is *partly satisfied* and has the possibility of being violated.

3   Problem Statement

Recently, the revolution of business always causes the dynamic changes of workflow processes. When workflows are more and more complex, it is necessary to develop the ability of analyzing workflows when they are edited. One characteristic of this ability is incremental analysis.

If the TWF-nets are small, the designers can easily obtain the checking results of temporal constraints after changes through the computing and analysis procedure, as discussed in [9]. However, in reality a large scale workflow may contain hundreds or thousands of data-intensive and computation-intensive activities or sub-processes. Unfortunately, the construction of sprouting graphs of TWF-nets suffers from exponential complexity. Hence, the checking work is inefficient in terms of time and space, because it needs to repeatedly compute the time information of all possible route nets of transaction instances in the processes, especially when the TWF-nets are changed frequently at build-time.

Based on the above discussion, the formal statement of problems we focus on is described as follows: Given a TWF-net W which contains multiple processes $\{W_1, W_2,\ldots, W_n\}$, there are some temporal constraints like $D_R(t_i, t_j) \leq s$, where transition t_i and t_j may belong to different processes in W. When some changes of W are made in an incremental way, how can we efficiently maintain the sprouting graphs of TWF-nets without analyzing the whole TWF-nets and check the potential temporal violations?

4   Obtaining Change Region of TWF-nets

Given a specific change on a TWF-net, we can locate the regions of the old and new TWF-nets containing all the transitions and places affected by the changes directly or indirectly [2].

Definition 4. (New and old change regions): Assume the old TWF-net before the change is $TWF_1=(P_1, T_1, F_1, FI_1)$, the new TWF-net after the change is $TWF_2=(P_2, T_2, F_2, FI_2)$, and the set of changes nodes is N_k. The new change region in TWF_2 is defined as $TWF_{2k}^c =(P_{2k}^c, T_{2k}^c, F_{2k}^c)$, where $P_{2k}^c =P_2 \cap N_k$; $T_{2k}^c = T_2 \cap N_k$; $F_{2k}^c =((P_{2k}^c \times T_{2k}^c) \cup (T_{2k}^c \times P_{2k}^c)) \cap F_2$. The old change region in TWF_1 is defined as $TWF_{1k}^c =(P_{1k}^c, T_{1k}^c, F_{1k}^c)$, where $P_{1k}^c = {}^\bullet T_{1k}^c \cap T_{1k}^{c\,\bullet}$; $T_{1k}^c =T_1 \backslash (T_2 \backslash T_{2k}^c)$; $F_{1k}^c =((P_{1k}^c \times T_{1k}^c) \cup (T_{1k}^c \times P_{1k}^c)) \cap F_1$.

Algorithm 1: Construct Change Regions
Input: The old and new TWF-nets TWF_1 and TWF_2
Output: The new change and old change regions
Step 1: Compare the flow relations of TWF_1 and TWF_2, and set $NS=\{x, y| (x, y) \in ((F_2 \backslash F_1) \cup (F_1 \backslash F_2))\} \cap T_2$, and $NT=\{x| FI_1(x) \neq FI_2(x)\}$ Then, we obtain the set $N= NS \cup NT$.

Step 2: For every transition $t \in N$, obtain its $^\bullet t$ and $t^\bullet$ in TWF_2, and $N \leftarrow \{^\bullet t \cup t^\bullet\}$. Partition N into $N_1, N_2,...,N_m$ such that $\forall k, l \in \{1,..., m\}, k \neq l: N_k \cap N_l = \varnothing$.

Step 3: For every N_k repeat the following sub-steps:

$P_{ik} = \varnothing, P_{ok} = \varnothing$;

Set the previous neighbor places of nodes in N_k is $Pre = \{p | ^\bullet p \cap N_k \neq \varnothing \wedge p^\bullet \cap N_k = \varnothing\}$; $Post = \{p | p^\bullet \cap N_k \neq \varnothing \wedge ^\bullet p \cap N_k = \varnothing\}$; If $\exists p_x \in Pre$: $\forall p \in N_k$, there exists a path from p to p_1, then $p_x \rightarrow P_{ik}$; If $\exists p_y \in Post$: $\forall p \in N_k$, there exists a path from p to p_2, then $p_y \rightarrow P_{ok}$;

For every $p \in N_k \backslash P_{ik}, N_k = N_k \cup ^\bullet p$; For every $p \in N_k \backslash P_{ok}, N_k = N_k \cup p^\bullet$; For every $t \in N_k, N_k = N_k \cup ^\bullet t \cup t^\bullet$; If $\exists l \in \{1,...,m\}, l \neq k; N_k \cap N_l \neq \varnothing$ then merge N_k and N_l as $N_{kl} = N_k \cup N_l$; So that, we get the new change regions $TWF_{21}^c \dots TWF_{2k}^c$.

Step 4: Based on obtained change regions $TWF_{21}^c \dots TWF_{2k}^c$, we get the uninfluenced parts $\{TWF_2 - TWF_{21}^c \dots TWF_{2k}^c\}$. Then, the old change regions $TWF_{11}^c \dots TWF_{1k}^c$ are obtained by deleting the uninfluenced parts from TWF_1.

The key of Algorithm 1 is to get the different parts between the old and new TWF-nets by comparing them. Then, we locate the temporal source and sink places. The extension is repeated until no more adjacent nodes are added. During the extension, if two newly generated components share some common nodes, then they should be merged. Assume that there are n_1 and n_2 transitions in TWF_1 and TWF_2, The time complexity of the Algorithm 1 is $O(n_1 n_2)$.

5   Maintaining Sprouting Graphs and Analyzing Temporal Constraints

The procedure of maintaining sprouting graphs after changes is presented in Algorithm 2.

Algorithm 2: Maintaining Sprouting Graph
Input: the old sprouting graph *OSG*
Output: the updated sprouting graph *USG*
Step 1: Obtain all the new change regions $\{TWF_{21}^c \dots TWF_{2k}^c\}$ and old change regions $\{TWF_{11}^c \dots TWF_{1k}^c\}$ by comparing old and new TWF-nets TWF_1 and TWF_2.

Step 2: We compute the parts of sprouting graph $\{SPc_1,..., SPc_m\}$ for $\{TWF_{21}^c \dots TWF_{2k}^c\}$. Then, we use $\{SPc_1,..., SPc_m\}$ to substitute the parts of sprouting graphs of $\{TWF_{11}^c \dots TWF_{1k}^c\}$.

Step 3: Assume the set of process in TWF_1 having changes is $\{P_1, ..., P_n\}$, we take the first one denoted as Px, and delete it from the set. If the set is null, the Algorithm ends.

Step 4: If Px has no any resource dependencies with another processes, go to Step 3. If Px has resource dependency between t_i and t_j of another process Py, respectively, go to Step 5.

Step 5: We compute the time interval $[\text{EST}(t_j), \text{LET}(t_j)]$ based on the structure of TWF_2, which $\text{EST}(t_j)$ is the earliest start time and $\text{LET}(t_j)$ is the latest end time $\text{LET}(t_j)$. If the time interval is not affects by changes, go to Step 3. Otherwise, we update the nodes affected in OSG, go to Step 3.

Step 6: Return the updated sprouting graph denoted as USG.

Assume that there are n transitions and m places in change regions of TWF_2, the time complexity of the Algorithm 2 is $O(nm^2)$.

To check the temporal constraint $D_R(t_i, t_j) \leq s$ by the updated sprouting graph, we reuse the checking procedure in our earlier work [9]. If there is any temporal validation, we return the paths information in the sprouting graphs to guide the designers to modify them.

6   Case Study

6.1   Case Specification

The production of a cell phone is mainly composed of two processes, i.e., production of electronic main board and production of peripheral parts. The first process is composed of requirement analysis to final production. The second one is composed of marketing to final production. The two processes execute concurrently. And there are two activities in each process which share a human resource (head engineer).

These two processes are specified as the Fig.1. Two transitions t_{18} and t_{24} share a human resource (the head engineer) represented by P_{Rl}. The transitions for these activities and corresponding meanings in the processes are shown in Table 2. Two following temporal constraints are specified.

(1) $D_R(t_{23}, t_{25}) \leq 20$: Manufacturing of peripheral parts ends no more than 20 time units after writing a technical document begins.

(2) $D_R(t_{23}, t_{19}) \leq 9$: Manufacturing of electronic main board ends no more than 9 time units after writing a technical document of peripheral parts begins.

Table 2. The activities of two workflow processes

Activities	Transitions	Activities	Transitions
analyze requirement	t_{11}	marketing	t_{21}
write a requirement document	t_{12}	feasibility analysis	t_{22}
design software	t_{14}	write technical document	t_{23}
design electrical circuit part	t_{15}	approved	t_{24}
reconstruct based on other models	t_{16}	manufacture	t_{25}
approved	t_{18}	and–split	t_{13}
manufacture	t_{19}	and–join	t_{17}

Based on [9], we construct the sprouting graph as shown in Fig.2 and Table 1. Note that there are no any temporal violations at the current state for the appropriate design of the activities of the logic and execution time.

6.2   Analysis in the Incremental Way

After some time, in order to quicken the speed to release products into the market, the designers are forced to change their processes with holding all temporal constraints. In an incremental way the designers may try some possible changes of models to attain the business goal. In order to simplification, we assume these two processes only are changed one time, and two parts of them are made as shown in Fig.3.
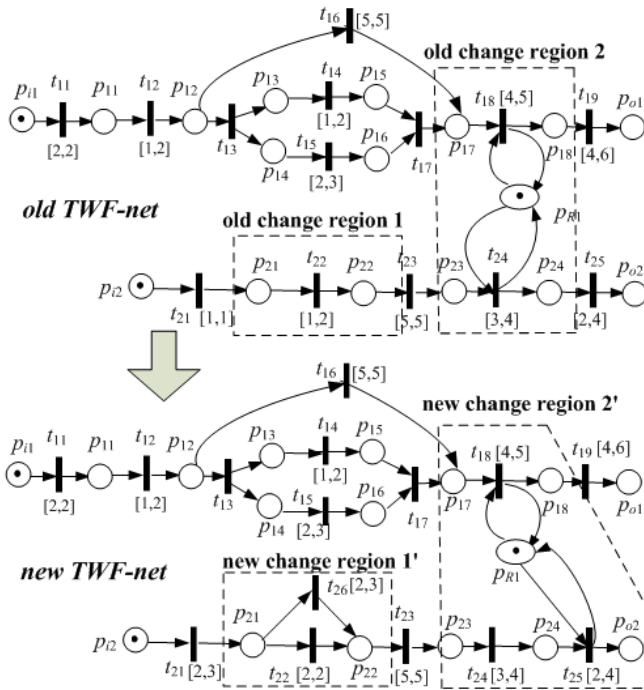
Fig. 3. The new TWF-net after changes and its updated sprouting graph

Firstly, we obtain two change regions 1 and 2 in the old TWF-net, and two corresponding change regions 1' and 2' in the new TWF-net as shown in Fig. 3. Secondly, using Algorithm 1, we maintain the sprouting graph as shown in Fig. 4, where the dark nodes are updated and the rest ones remain unchanged with Fig. 2. The time and path sets of whose updated nodes are shown in Table 3.

(1) For $D_R(t_{23}, t_{25}) \leq 20$: we get that the time set of input nodes of arc corresponding to t_{23} is {[4,5], [4,6]} and the time set of the output nodes of arc corresponding to t_{25} is {[14,18], [14,19]}, respectively. Then, four ordered pairs are obtained: <[4,5],

[14,18]>,< [4,5], [14,19]>, <[4,6], [14,18]> and <[4,6], [14,19]>. For <[4,5], [14,18]>, we have $X=20$, $Y=[14-5,18-4]=[9,14]$. $D_R(t_{23},t_{25})\leq20$ is completely satisfied under the current ordered pair because of $20>14$. In similar way, the other three ordered pairs are checked as completely satisfied. Therefore, we know that $D_R(t_{23},\ t_{25})\leq20$ is satisfied.

(2) For $D_R(t_{23},\ t_{19})\leq9$: we get that $D_R(t_{23},\ t_{19})\leq9$ is partly satisfied and has the possibility of being violated in similar way. Further, the paths with violations, i.e., $\{\{t_{21},t_{22}\},\ \{t_{21},t_{22}\}\}$ and $\{t_{11},t_{12},t_{16}(t_{21},t_{22},t_{23}),t_{18},\ t_{19}\}$, are detected and reported to designers. Methods are needed to deal with the temporal violations after changes happened [9].
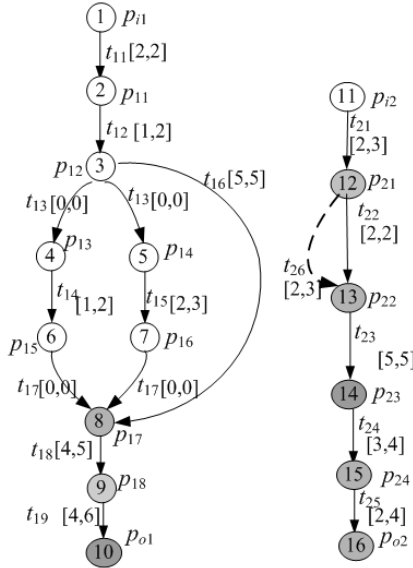
Fig. 4. The new TWF-net after changes and its updated sprouting graph

Table 3. The updated nodes of sprouting graph

Node	Time sets	Path sets
8	$\{[5,7],[8,9]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14})\ \|\ (t_{11},t_{12},t_{13},t_{15}),t_{17}\},\ \{t_{11},t_{12},t_{16}\}\}$
9	$\{[9,12],[12,14]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14})\ \|\ (t_{11},t_{12},t_{13},t_{15}),t_{17}(t_{21},t_{22},t_{23},t_{24}),t_{18}\},$ $\{t_{11},t_{12},t_{16}(t_{21},t_{22},t_{23}),t_{18}\}\}$
10	$\{[13,18],[16,20]\}$	$\{\{(t_{11},t_{12},t_{13},t_{14})\ \|\ (t_{11},t_{12},t_{13},t_{15}),t_{17}(t_{21},t_{22},t_{23},t_{24}),t_{18},t_{19}\},$ $\{t_{11},t_{12},t_{16}(t_{21},t_{22},t_{23}),t_{18},t_{19}\}\}$
12	$\{[2,3]\}$	$\{\{t_{21}\}\}$
13	$\{[4,5],\ [4,6]\}$	$\{\{t_{21},t_{22}\},\ \{t_{21},t_{22}\}\}$
14	$\{[9,10],\ [9,11]\}$	$\{\{t_{21},t_{22},t_{23}\},\ \{t_{21},t_{22},t_{23}\}\}$
15	$\{[12,14],[12,15]\}$	$\{\{t_{21},t_{22},t_{23},t_{24}\},\{\ t_{21},t_{22},t_{23},t_{24}\}\}$
16	$\{[14,18],[14,19]\}$	$\{\{\ t_{21},t_{22},t_{23},t_{24}((t_{11},t_{12},t_{13},t_{14})\ \|\ (t_{11},t_{12},t_{13},t_{15}),t_{17}),t_{25}\},$ $\{\ t_{21},t_{22},t_{23},t_{24}(t_{11},t_{12},t_{16}),t_{25}\}\}$

7   Comparison and Validation

7.1   Comparison by Several Test Cases

Compared with the approach in [9], our approach can effectively maintain the sprouting graph to the new state, instead of re-constructing a new one. In the worst case where the changes are made at the initial part of the TWF-nets, the entire sprouting graph will have to be recomputed, and this is identical with the approach in [9], i.e., having the same number of nodes to be recomputed. In the best case where the changes are made at the end of the graph, only the latter part of sprouting graph will have to be recomputed. In a real application there are often some changes made in the middle parts and thus we can expect it can always outperform the approach in [9].

In order to quantitatively analyzing performance of our approach, we compare it with the existing approach [9] by several test cases in Table. 4. There test cases are based on TWF-nets and randomly modified on some parts (regions). The criterion of comparison is the number of nodes in sprouting graph needed to update after changes. As we can see, our approach dramatically decreases the complexity of analyzing. Especially, with the increasing number of services increasing and the augmenting complexity of models, our approach is more efficient than the approach in [9].

Table 4. Comparison with the approach in [9]

Test cases with random modification	Approach in [9]	Our approach
1 process (10 places, 14 transitions , 0 resource places)	10 nodes	2 nodes
2 processes (16 places, 14 transitions, 1 resource places)	16 nodes	8 nodes
3 processes (27 places, 23 transitions, 0 resource places)	27 nodes	10 nodes
3 processes (28 places, 27 transitions, 2 resource places)	28 nodes	11 nodes
4 processes (34 places, 30 transitions, 0 resource places)	34 nodes	11 nodes
4 processes (45 places, 36 transitions, 2 resource places)	45 nodes	13 nodes

Note that we can check more than one temporal constraint by using the same updated sprouting graph. The detailed example is illustrated in the Section 6. Thus, our approach is scalable, i.e., we do not have to create a new sprouting graph for checking a new temporal constraint. Our approach has higher efficiency especially when it deals with large numbers of temporal constraints.

7.2   Validation by Uppaal

We use Uppaal [11] to validate our approach, which is an integrated tool for model checking of real-time systems, and its typical application areas include real-time controllers and communication protocols.

Firstly, we transform the TWF-nets to equivalent Timed Automata (TA) model [12]. For a transition, we define a TA with two locations *disabled* and *enabled*, one local clock x, and some integer parameters. Secondly, we construct two TA observers in which the temporal constraints are denoted as the guard conditions labeled with the

corresponding edges. Thirdly, both the transformed TA models and queries for temporal constraints are input into the model checking engine of Uppaal. Finally, the results are returned to indicate whether the temporal constraints are satisfied.

For each case in Table 4, the results of Uppaal are consistent with the results from our approach, so that the correctness of our solution is proved.

8   Conclusion

Incremental analysis gives designers some warnings or messages which assist them right after they make changes. In this paper, we present an incremental approach to check temporal violations. Our approach can dynamically check the temporal constraints of multiple workflow processes with resource dependencies, and considers the complex situations of modifying TWF-nets more than simply editing operations and can check the temporal violations efficiently. Furthermore, this approach is applicable and efficient in terms of time and space.

In the future, we would like to extend the research on the following issues: 1) At the run-time of workflow instances, how to handle the model changes of TWF-nets needs further research. 2) In some cases a resource is allowed to execute concurrently.

References

1. Hauser, R.F., Friess, M., Kuster, J.M., Vanhatalo, J.: An Incremental Approach to the Analysis and Transformation of Workflows using Region Trees. IEEE Trans. Syst., Man, Cybern.: Part C 38(3), 347–359 (2008)
2. Sun, P., Jiang, C.: Analysis of Workflow Dynamic Changes based on Petri Net. Information and Software Technology 51(2), 284–292 (2009)
3. Ellis, C., Keddara, K., Rozenberg, G.: Dynamic Change with in Workflow System. In: International Conference on Organizational Computing Systems, pp. 10–22. ACM Press, Milpitas (1995)
4. vander Aalst, W.M.P., Basten, T.: Inheritance of Workflows: An Approach to Tackling Problems Related to Change. Theoretical Computer Science 270(12), 125–203 (2002)
5. Eder, J., Panagos, E., Rabinovich, M.: Time Constraints in Workflow Systems. In: Jarke, M., Oberweis, A. (eds.) CAiSE 1999. LNCS, vol. 1626, pp. 286–292. Springer, Heidelberg (1999)
6. Chen, J., Yang, Y., Chen, T.Y.: Dynamic Verification of Temporal Constraints On-the-Fly for Workflow Systems. In: 11th Asia-Pacific Conference on Software Engineering, pp. 134–144. IEEE Press, New York (2004)
7. Chen, J., Yang, Y.: Temporal Dependency for Dynamic Verification of Fixed-Date Constraints in Grid Workflow Systems. In: Zhang, Y., Tanaka, K., Yu, J.X., Wang, S., Li, M. (eds.) APWeb 2005. LNCS, vol. 3399, pp. 820–831. Springer, Heidelberg (2005)

8. Li, H., Yang, Y.: Dynamic Checking of Temporal Constraints for Concurrent Workflows. Electronic Commerce Research and Applications 4(4), 124–142 (2005)
9. Du, Y., Xiong, P., Fan, Y., Li, X.: Dynamic Checking and Solution to Temporal Violations In Concurrent Workflow Processes. IEEE Trans. Syst., Man, Cybern.: Part A 41(6), 1166–1181 (2011)
10. Jong, W.T., Shiau, Y.S., Horng, Y.J., Chen, H.H., Chen, S.M.: Temporal Knowledge Representation and Reasoning Techniques Using Time Petri Nets. IEEE Trans. Syst., Man, Cybern.: Part A 29(4), 541–545 (1999)
11. Uppaal, `http://www.it.u.se/research/group/darts/uppaal`
12. Gruhn, V., Laue, R.: Using Timed Model Checking for Verifying Workflows. In: 2nd International Workshop on Computer Supported Activity Coordination, pp. 75–88. INSTICC Press (2005)

Keywords Filtering over Probabilistic XML Data

Chenjing Zhang[1,2], Le Chang[3], Chaofeng Sha[2]
Xiaoling Wang[3], and Aoying Zhou[3]

[1] College of Information Technology, Shanghai Ocean University, China
cjzhang@shou.edu.cn
[2] School of Computer Science, Fudan University, China
{cjzhang, cfsha}@fudan.edu.cn
[3] Shanghai Key Laboratory of Trustworthy Computing,
Software Engineering Institute, East China Normal University
{xlwang,ayzhou}@sei.ecnu.edu.cn,changle@live.cn

Abstract. Probabilistic XML data is widely used in many web applications. Recent work has been mostly focused on structured query over probabilistic XML data. A few of work has been done about keyword query. However only the independent and the mutually-exclusive relationship among sibling nodes are discussed. This paper addresses the problem of keyword filtering over probabilistic XML data, and we propose PrXML$^{\{exp,ind,mux\}}$ model to represent a more general relationship among XML sibling nodes, for keywords filtering over probabilistic XML data. $kdptab$ is defined as keyword distribution probability table of one subtree. The Dot product, Cartesian product, and addition operation of $kdptab$ are also defined. In PrXML$^{\{exp,ind,mux\}}$ model, XML document is scanned bottom-up and achieve keyword filtering based on SLCA semantics efficiently in our method. Finally, the features and efficiency of our method are evaluated with extensive experimental results.

Keywords: Probabilistic XML, Keywords Filtering, SLCA.

1 Introduction

Many important XML applications such as automatic information integration and fault diagnose produce uncertainty. Probabilistic XML data management is becoming a critical issue. In the current probabilistic XML model [1,2,3,4,5,6], a probabilistic XML document(p-$document$) is considered as a labelled tree, consisting of two types of nodes, $ordinary$(ORD) nodes representing actual data and $distributional$ nodes representing the probability distribution of the child nodes. $Distributional$ nodes have five types, $\{IND, MUX, DET, EXP, CIE\}$. [5] defines p-$document$ family as PrXML^C,$C \subseteq \{IND, MUX, DET, EXP, CIE\}$. [7,8] have discussed structured query over probabilistic XML data. Only [9] concerns with keyword query, in which two types of $distributional$ nodes, IND and MUX(the children of them are independent and mutually exclusive respectively), are considered.

From the XML keyword search aspect, SLCA [10,11] is a widely accepted keyword semantics. SLCA-based keywords filtering over probabilistic XML data meets new challenges, such as how to compute the probability for a certain node to be a SLCA node and whether a SLCA node is still SLCA node and so on.

To solve these problems, we propose a SLCA-based keywords filtering solution over probabilistic XML data model $PrXML^{\{exp,ind,mux\}}$.

We summarize the contributions of this paper as follows:

- We present a more general model $PrXML^{\{exp,ind,mux\}}$ for keywords filtering over probabilistic XML data.
- Algorithms are given to find SLCA nodes and compute the probability for a node to be a SLCA node using *kdptabs*, which are keyword distribution probability table of subtrees. Dot product, Cartesian product and addition operation of *kdptab* are also defined for probability computations.
- Experimental evaluation has demonstrated the efficiency of the proposed filtering algorithm.

The rest of the paper is organized as follows. Motivations and preliminary knowledge will be given in Section 2. Section 3 gives the overview of the keyword filtering over probabilistic XML. The system architecture and algorithms are proposed in Section 4. Section 5 evaluates the features and efficiency of our algorithms through experiments. Conclusions and future work are in the end.

2 Motivations and Preliminaries

2.1 Motivations

Some common and useful data dependencies deserve investigation. For instance, both authors, $Tommy$ and $Hung$, in an XML document shown in Fig. 1 are uncertain. If $Tommy$ appears, $Hung$ appears with some probability(for example cooperation relationship). This kind of data dependency is common among XML data and has great merits in maintaining data accuracy and query correctness. The sibling dependency can be described by EXP node, which can represent arbitrary sibling nodes dependency relationship using node sets.

However, [9] didn't discuss the probability model including EXP nodes. This paper focuses on keyword filtering over model $PrXML^{\{exp,ind,mux\}}$, which represents more general relationship among sibling nodes. Following are some definitions related to our filtering algorithms.

2.2 Preliminaries

Example 1. *keyword query: $Tommy, 2008$.*

Example 1 gives a keyword query. Fig. 1 shows a probabilistic XML tree. The tree contains *ordinary* nodes, EXP nodes and MUX nodes. The number attached on each edge in the tree indicates child node's conditional probability
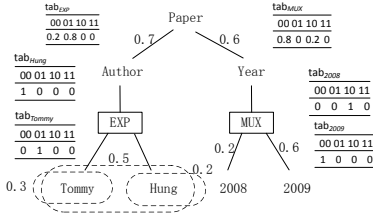
Fig. 1. a Probabilistic XML Tree
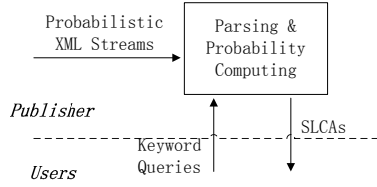
Fig. 2. Keywords Filtering over Probabilistic XML

given the existence of the parent node. "1" is the default conditional probability. The node EXP in Fig. 1 has three children sets, $\{Tommy\}$, $\{Hung\}$ and $\{Tommy, Hung\}$, whose set probabilities are 0.3, 0.2 and 0.5 respectively. IND nodes are omitted in Fig. 1 because computing the probability of IND nodes and of *ordinary* nodes are the same [9].

Definition 1. *The kdptab of subtree $SubT(n)$, namely $kdptab_n$, is defined in equation 1.*

$$
\begin{aligned}
kdptab_n &= \{item_i | 0 \le i \le 2^{KeywordNumber} - 1\}. \\
item_i &=< bitvector, Prob > .
\end{aligned}
\tag{1}
$$

bitvector is a binary string which record which keywords appear. $kdptab_n$ is used to maintain the keyword distributions of the subtree $SubT(n)$ rooted at node n. Take node $Tommy$ as an example. $kdptab_{Tommy}$ is shown in Fig. 1. One term $< "01", 1 >$ in $kdptab_{Tommy}$ means that only the first keyword in Example 1 appears in $SubT(Tommy)$ and its probability is 1. $kdptab_n$ is defined in Definition 1.

Definition 2. *Dot product($\cdot$) of kdptab:*

$$
\begin{aligned}
kdptab \cdot p = \{< bitvector, Prob > \;|\; &< bitvector, Prob_0 >\in kdptab, \\
&Prob = Prob_0 * p\}
\end{aligned}
\tag{2}
$$

Dot product of *kdptab* is defined in Definition 2. Take $kdptab_{2008}$ in Fig. 1 as an example. $kdptab_{2008} \cdot 0.2 = \{< "00", 0 >, < "01", 0 >, < "10", 0.2 >, < "11", 0 >\}$.

Definition 3. *Addition operation($+$) of kdptab:*

$$
\begin{aligned}
kdptab_1 + kdptab_2 &= S_1 \cup S_2 \cup S_3, \\
S_1 =& \{< Bt, Prob_{item_i} + Prob_{item_j} > \;|\; < Bt, Prob_{item_i} >\in kdptab_1 \wedge \\
& < Bt, Prob_{item_j} >\in kdptab_2\}, \\
S_2 =& \{< Bt, Prob_{item_i} > \;|\; < Bt, Prob_{item_i} >\in kdptab_1 \wedge \\
& (\forall j, item_j \in kdptab_2 \wedge Bt \ne bitvector_{item_j})\}, \\
S_3 =& \{< Bt, Prob_{item_j} > \;|\; (\forall i, item_i \in kdptab_1 \wedge Bt \ne bitvector_{item_i}) \wedge \\
& < Bt, Prob_{item_j} >\in kdptab_2\}.
\end{aligned}
\tag{3}
$$

Definition 3 gives the addition operation of *kdptab*. For example, given $kdptab_1 = kdptab_{2009} \cdot 0.6$, $kdptab_2 = kdptab_{2008} \cdot 0.2$, $kdptab_1 + kdptab_2 = \{<$ "00", $0.6 >, < $ "01", $0 >, < $ "10", $0.2 >, < $ "11", $0 >\}$.

Cartesian product of *kdptab* is described in Definition 4.

Definition 4. *Cartesian product($\times$) of kdptab*

$$kdptab_1 \times kdptab_2 = \{item_i \times item_j | \; \forall \; item_i \in kdptab_1 \wedge \forall item_j \in kdptab_2\}. \quad (4)$$

$$item_i \times item_j = \{< newBt, newProb > | \; newBt = bitvector_{item_i} | \; bitvector_{item_j}$$
$$\wedge \; newProb = Prob_{item_i} * Prob_{item_j}\}.$$
$$(5)$$

In Fig. 1, $kdptab_{Author}$ equals $kdptab_{EXP}$ and $kdptab_{Year}$ equals $kdptab_{MUX}$. Given $kdptab_1 = kdptab_{Author} \cdot 0.7$ and $kdptab_2 = kdptab_{Year} \cdot 0.6$, the result of $kdptab_1 \times kdptab_2$ equals $kdptab_{Paper}$, which is shown in Table 4.

Definition 5 gives the priority of *kdptab*'s operations.

Definition 5. *The operation priority of $+, \times$ and $\cdot$ increases successively.*

Lemma 1. *For any $kdptab_1$, $kdptab_2$ and $kdptab_3$, $kdptab_1 \times kdptab_3 + kdptab_2 \times kdptab_3 = (kdptab_1 + kdptab_2) \times kdptab_3$.*

The detail proof for this lemma is omitted due to limited space.

3   SLCA Computation over Probabilistic XML Data

In a deterministic XML document, a node v is a SLCA node if (1) the subtree $SubT(v)$ rooted at the node v contains all the keywords, and (2) there doesn't exist any v's descendant satisfies condition(1). [9] mentioned that a SLCA node is represented by a 2-tuple $< v, p >$ in probabilistic XML. v is a document node and p is the probability for v to be a SLCA node. p can be computed by equation 6.

$$Prob_{SLCA} = Pr(path_{r \to v}) * Pr_{local}(v). \quad (6)$$

$Pr(path_{r \to v})$ indicates the existence probability of v in all possible worlds. $Pr_{local}(v)$ is the containing probability that $SubT(v)$ contains all keywords.

3.1   Existence Probability $Pr(path_{r \to v})$

Bayes equation can be used to compute nodes' existence probability in probabilistic XML [2]. One node's existence probability is only related to the conditional probabilities of all the nodes on the path from the root node to the current node. However, EXP nodes gives conditional probabilities of child node sets instead of conditional probabilities of each individual child node. Then we'll discuss how to compute node v's existence probability when EXP nodes exist.

If all EXP nodes appear in the descendants of the current node v, then every conditional probability of the nodes on the path$(r \to v)$ is known. The existence probability of v can be computed by multiplying every conditional probability of the nodes on the path. Otherwise, If there is any EXP node on the path from the root node r to the current node v, one of the EXP node's child node c is also on the path. c's conditional probability should be computed from sets' conditional probabilities firstly. Then v's existence probability can be computed. For example, there's an EXP node on the path from root $Paper$ to node $Tommy$ in Fig. 1. We compute node $Tommy$'s conditional probability using the set probabilities. We can get the existence probability of node $Tommy$.

3.2   Containing Probability $Pr_{local}(v)$

Given the condition that no descendants of node v are SLCA nodes, the probability for node v to be a SLCA node equals the total probability of $SubT(v)$'s possible worlds including all keywords. $kdptab_v$ is used to record the probability for $SubT(v)$ to contain query keywords. Following we'll describe how to use tuples from children to get $kdptab_v$.

Tuples from Children to Parents: Each child node transmit to parent node a 2-tuple $< prob, kdptab >$. $Prob$ is the conditional probability of the child node and $kdptab$ is the keyword distribution probability table.

Example 1 is an keyword query. In Fig. 1, the 2-tuple transmitted by $Tommy$ to its parent node is $< 0.8, kdptab_{Tommy} >$ and $2008 < 0.2, kdptab_{2008} >$. 0.8 and 0.2 are node $Tommy$ and node 2008's conditional probability respectively. Terms whose $prob$ is non-zero are recorded to save storage space.

Tuples Processing: Normally, any none-leaf node v processes tuples from children following three steps: (1)Compute probabilities of possible worlds which are composed of v's child nodes based on the relationships among child nodes. (2)Compute the keyword distribution probability for each possible world using $kdptab$ from children. (3)Sum correlative probabilities of all possible worlds. Then we can obtain $kdptab$ of $SubT(v)$. However, the above steps scan nodes multiple times. Following we'll try to merge above steps for IND(and ORD), MUX, EXP nodes to reduce time complexity.

Case 1- MUX: M is a *distributional* node which has n mutually-exclusive children $X_1, X_2, \ldots, X_n$, and P_i is the conditional probability of X_i. $kdptab_{x_i}$ is the keyword distribution probability table for $SubT(X_i)$ when node X_i exists. We define P_{n+1} as the probability for M to have no children. Then $P_{n+1} = 1 - \sum_{1 \leq i \leq n} P_i$ and $kdptab_{n+1} = \{< \text{"00"}, 1 >\}$ is the keyword distribution table of this possible world. The keyword distribution table $kdptab_M$ can be computed by Equation 7. Using Equation 7, MUX node's keyword distribution table can be obtained after scanning child nodes once.

$$kdptab_M = \sum_{1 \leq i \leq n+1} kdptab_{X_i} \cdot P_i \qquad (7)$$

$kdptab_{MUX}$(shown in Fig. 1) of node MUX can be obtained using $kdptab_{2008}$ and $kdptab_{2009}$ according to Equation 7.

Case 2- EXP: Suppose EXP node E has m child node subsets, which corresponds to $SubT(E)$'s m possible worlds. The existence probability for i^{th} subset is P_i and the subset contains n child nodes whose $kdptab$ information $kdptab_i(1 \le i \le n)$ are already obtained. Then the keywords distribution table for i^{th} possible world $kdptab_{pw_i}$ can be computed by Equation 8.

$$kdptab_{pw_i} = kdptab_1 \times kdptab_2 \times \cdots \times kdptab_n. \tag{8}$$

If $kdptab$ of each possible world is $kdptab_{pw_1}, kdptab_{pw_2}, \cdots, kdptab_{pw_m}$, $kdptab_E$ can be computed by Equation 9.

$$kdptab_E = P_1 \cdot kdptab_{pw_1} + P_2 \cdot kdptab_{pw_2} + \cdots + P_m \cdot kdptab_{pw_m}. \tag{9}$$

In Fig. 1, node EXP has 3 child subsets:$\{Tommy\}, \{Hung\}, \{Tommy, Hung\}$. $kdptabs$ of $SubT(EXP)$'s corresponding possible worlds are shown in Table 1, Table 2 and Table 3. Then $kdptab_{EXP}$ can be obtained using Equation 9.

Table 1. $kdptab_{pw_1}$ **Table 2.** $kdptab_{pw_2}$ **Table 3.** $kdptab_{pw_3}$ **Table 4.** $kdptab_{Paper}$

00	01	10	11
0	1	0	0

00	01	10	11
1	0	0	0

00	01	10	11
0	1	0	0

00	01	10	11
0.3872	0.4928	0.0528	0.0672

Case 3- IND and ORD: Suppose there is an IND or ORD node. It has n independent children $X_1, X_2, \ldots, X_n$, whose joint distribution is:

$$P(X_1, X_2, \ldots, X_n) = P(X_1)P(X_2) \cdots P(X_n). \tag{10}$$

There are 2^n possible worlds, whose keyword distribution probability table can be obtained by Equation 11.

$$P(X_1) \cdot kdptab_{X_1} \times P(X_2) \cdot kdptab_{X_2} \times \cdots \times P(X_n) \cdot kdptab_{X_n} \tag{11}$$

X_i's existence probability $P(X_i = 1)$ is given. $P(X_i = 0) = 1 - P(X_i = 1)$. If X_i exists, $SubT(X_i)$'s kdptab information $kdptab_{X_i=1}$ is already computed using children's $kdptabs$. If X_i doesn't exist, no nodes are in $SubT(X_i)$. So its kdptab information $kdptab_{X_i=0} = \{< \text{"00"}, 1 >\}$. Using $kdptab'_{X_i}$ to represent $P(X_i) \cdot kdptab_{X_i}(1 \le i \le n)$, Equation 11 equals Equation 12.

$$kdptab'_{X_1} \times kdptab'_{X_2} \times \cdots \times kdptab'_{X_n} \tag{12}$$

Then we get $kdptab_{IND}$ by adding $kdptabs$ of each possible world.

$$kdptab_{IND} = \sum_{X_1, X_2, \ldots, X_n} kdptab'_{X_1} \times kdptab'_{X_2} \times \cdots \times kdptab'_{X_n}. \tag{13}$$

It is easy to prove that Equation 13 equals Equation 14 using Lemma 1:

$$kdptab_{IND} = \sum_{X_1} kdptab'_{X_1} \times \sum_{X_2} kdptab'_{X_2} \times \cdots \times \sum_{X_n} kdptab'_{X_n}. \qquad (14)$$

An ORD node may contain keywords itself. So Equation 14 needs to be altered to Equation 15 ($kdptab_{ORD_{local}}$ records the keyword distribution probabilities of the ORD node) to compute $kdptab_{ORD}$. We can get the kdptab information of IND and ORD nodes using Equation 14 and Equation 15 by scanning their child nodes once.

$$kdptab_{ORD} = kdptab_{ORD_{local}} \times \sum_{X_1} kdptab'_{X_1} \times \sum_{X_2} kdptab'_{X_2} \times \cdots \times \sum_{X_n} kdptab'_{X_n}.$$
$$(15)$$

In Fig. 1, node $Paper$ is an ORD node. $kdptab_{Author} = kdptab_{EXP}$, $kdptab_{Year} = kdptab_{MUX}$. $kdptab_{EXP}$ and $kdptab_{MUX}$ are shown in Fig. 1. $kdptab_{Paper} = (0.7 \cdot kdptab_{Author} + 0.3 \cdot \{< \text{“00”}, 1 >\}) \times (0.6 \cdot kdptab_{Year} + 0.4 \cdot \{< \text{“00”}, 1 >\})$. $kdptab_{Paper}$ is shown in table 4. The 2-tuple $< \text{“11”}, 0.0672 >$ in Table 4 means that the probability for $SubT(Paper)$ to contain both keywords is 0.0662. So the probability for the node $Paper$ to be a SLCA node is 0.0672.

4   Implementations

Fig. 2 shows the system architecture of SLCA-based keywords filtering over probabilistic XML. Users submit keyword queries. The system receives probabilistic XML data continually, parses these data and returns results based on users' queries. The results are in the form of SLCA nodes.

We use SAX parser to read XML document. The parser will produce a stream of events. The main events include $StartDocument$, $EndDocument$, $StartElement$, $EndElement$ and $Characters$. We keep a record ($level, id, bitvector, nodeProb, type, expInfo, kdptab$) for every XML node (whether it is $ordinary$ node or $distributional$ node). $Level, id$ represent the depth and ID of the current node. $Bitvector$ are the keywords inclusion information. $NodeProb$ records the conditional probability of the current node. $Type$ represents the type of the current node and its value can be ORD, IND, MUX, EXP or SLCA. $Type = \text{“SLCA”}$ means that either the current node or at least one descendent of it is a SLCA node and it will be transmitted upwards to prevent ancestors from being a SLCA node. $ExpInfo$ records the subsets information of child nodes. $kdptab$ is the keyword distribution probability table. In fact, we can consider SAX parsing as preorder traversal of XML tree. The records are transmitted bottom-up. After all descendants are processed, all information of children will be combined with information of their parent. Once SLCA node is identified, put ID of it into result-set. When the parsing is finished, all computation will end.

Algorithm 1. EndElement

Input:
   an element name.
Output:
 1: $info = Stack$.pop();
 2: **if** $info$.getLevel() $==$ level **then**
 3:    $nodeProb \leftarrow$ get nodeProb from $pInfo$;
 4:    $slcaProb$=isSLCA(info);
 5:    **if** $slcaProb$!=0 **then**
 6:       **if** isTopK(info,$slcaProb$) **then**
 7:          set $type$ of $info$ to be "SLCA";
 8:       **end if**
 9:    **end if**
10:    **if** $info.kdptab$ isn't $\emptyset$ or $info.type$ equals "SLCA" **then**
11:       $Stack$.push($info$);
12:    **end if**
13: **else**
14:    $pInfo \leftarrow$ get current node info from $Stack$;
15:    $nodeProb \leftarrow$ get nodeProb from $pInfo$;
16:    **if** current node is "IND" node or "ORD" node **then**
17:       INDChild($pInfo$);
18:    **end if**
19:    **if** current node is "MUX" node **then**
20:       MUXChild($pInfo$);
21:    **end if**
22:    **if** current node is "EXP" node **then**
23:       EXPChild($pInfo$);
24:    **end if**
25:    $slcaProb$=isSLCA(info);
26:    **if** $slcaProb$!=0 **then**
27:       **if** isTopK(pInfo,$slcaProb$) **then**
28:          set $type$ of $info$ to be "SLCA";
29:       **end if**
30:    **end if**
31:    **if** $info.kdptab$ is $\emptyset$ and $info.type$ not equals "SLCA" **then**
32:       $Stack$.pop($info$);
33:    **end if**
34: **end if**
35: $pathProb = pathProb$ / $nodeProb$;
36: $level =level$ -1;

4.1   Implementation

StartDocument event initializes some global variables. If the parser meets the beginning of an element, it triggers *StartElement* event. *StartElement* event constructs current element's record, which is "$level, id, bitvector, nodeProb, type, expInfo, kdptab$", and push it into *Stack*. Variable *pathProb* is used to record

current node's existence probability. *Bitvector* and *kdptab* of current node may be updated when *Characters* event comes.

End mark of an element triggers event *EndElement*(Algorithm 1). When *EndElement* event comes, we will combine the tuples from child nodes with it of the current node and compute the probability for the current node to be a SLCA node. If the current node is a leaf node (Line2-12), function *isSLCA()* is called to get the probability of current node to be SLCA node. If the probability is high enough(judged by function *isTopK()*), then the current node is a SLCA node and this information will be recorded in *type*. If the current node is not a leaf node (Line13-34), information of all the child nodes are popped from *Stack*. We'll obtain the record of current node by equations in subsection 3.2 according to the relationship among child nodes. If the subtree rooted at the current node contains all the keywords and the SLCA probability is enough high, then the current node is a SLCA node and this information will be recorded in *type*.

Only *ordinary* nodes can be SLCA nodes. Function *isSLCA()* makes sure that no descendants of the current node is a SLCA node and then confirms that the current node is an *ordinary* node. Afterwards function *isSLCA()* will read the probability of the node containing all the keywords from *kdptab*. The probability multiply *pathProb* and be returned as the probability for the current node to be a SLCA node.

4.2   Optimization

Line 10 and line 31 give optimizations just for the case that *p-document* has no EXP nodes. The condition shows that if the current node doesn't contain any keyword and its subtree doesn't have SLCA nodes, its record needn't to be stored in *Stack*. Temporarily, there is no optimization towards the *p-document* which contains EXP node.

Time complexity of the whole method is related to the number and the types of document nodes. ORD, IND and MUX nodes need to be scanned only once during the process. Child nodes of the EXP node, however, need to be scanned multiple times because they may appear in different subsets. Given a document which has n nodes, m_1 nodes are child nodes of EXP nodes. There are s sets composed of child nodes of EXP nodes, and on average there will be m_2 nodes in each set. In terms of the data size, the time complexity of the whole method is $O(n + s * m_2 - m_1)$. So, given a fixed size document, the number of subsets, the number of nodes in the subsets, and the number of EXP nodes will affect the algorithm efficiency.

5   Experiment

In this section, experiments are given to evaluate SLCA keywords filtering algorithms over probabilistic XML. The experiments analyze the features and efficiency of the algorithms from different aspects such as data set and query features. The program is coded in Java. The experiments are performed using

dual-core desktops with 2GB memory. We use real dataset DBLP.We generate probabilistic XML tree using the same method as used in [9,7]. We visit the nodes in the original XML tree in pre-order way. For each node v visited, we randomly generate some *distributional* nodes of "IND" or "MUX" or "EXP" types as children of v with the percentage of 5% - 20%. For the original children of v, we choose some of them as the children of the new generated *distributional* nodes. Then random probability distributions are assigned to these children with the restriction that the sum of them for a MUX node is no greater than "1" and the sum of them for a EXP node is "1". The existence probabilities of each individual child node of the EXP nodes should be pre-calculated and stored. The features of datasets used in this experiment are shown in Table 5.

Table 5. Dataset Features

ID	orig-size	size	#IND	#MUX	#EXP	#subset	#Ordinary
Doc1	5M	6.24M	7,016	7,366	0	0	130,047
Doc2	5M	6.6M	4,643	4,854	4,763	9,526	130,047
Doc3	10M	12.4M	13,492	14,173	0	0	251,581
Doc4	10M	13.1M	9,213	9,344	9,113	18,226	251,581
Doc5	30M	37.2M	41,325	42,437	0	0	765,889
Doc6	30M	39.4M	27,749	28,687	28,162	56,324	765,889
Doc7	50M	62.2M	69,383	72,166	0	0	1286,098
Doc8	50M	65.8M	46,841	48,489	46,415	92,830	1,286,098
Doc9	131M	160M	83,087	86,430	0	0	3,332,130
Doc10	131M	164M	55,825	57,390	55,648	111,296	3,332,130
Doc11	131M	164M	179,541	187,299	0	0	3,332,130
Doc12	131M	168M	121,172	124,653	121,268	242,536	3,332,130
Doc13	131M	173M	261,812	271,457	0	0	3,332,130
Doc14	131M	177M	176,228	181,438	175,624	351,248	3,332,130
Doc15	131M	180M	343,893	356,290	0	0	3,332,130

Queries in [9] are parted of queries used in our experiment. We also adopted the query generating pattern kN-L-H in [11]. kN is the number of keywords in the queries. L and H are frequencies of the lowest frequency and highest frequency of keywords. We randomly generate a group of queries and execute them 5 times. The average time of each query is recorded in following figures.

Ratio of *distributional* ***Nodes:*** There are four options, 5%, 10%, 14.5% and 20%, of the ratio of the *distributional* nodes. The corresponding document are *Doc*9, *Doc*11, *Doc*13 and *Doc*15. The queries used are the same with queries in [9]. Fig. 3 shows the average execution time of different queries over different datasets using optimized(just for no-EXP docs) or non-optimized algorithms. Experimental results indicate that the execution time increases linearly with the increasing of distributional nodes. Scanning and parsing the added data are the main reason for the time growth. The optimized algorithm not only saves storage space, but also avoids processing irrelevant nodes and boosts efficiency.
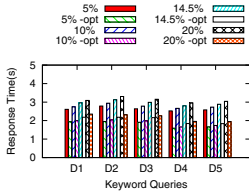
Fig. 3. *distributional* Node Ratio's Effect
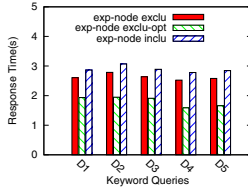
Fig. 4. Vary Query over Doc.9,10
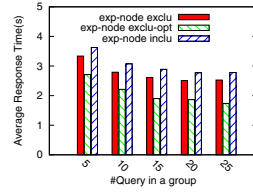
Fig. 5. Effect of #Query in a Query Group

Comparison with Previous Work: The experiment adopt similar datasets (considering the number of nodes, ratio of distributional nodes, size of the document) and same queries with them of [9]. And it makes comparison on query efficiency with [9]. Fig. 4 shows either for optimized or non-optimized algorithm, the execution time of the five queries ranges from 1.6 to 3.1 seconds. However, the execution time of the same query consumes about 7-13 seconds in [9]. Our algorithm compute the SLCA probability as soon as a SLCA node is identified and no redundant computations. [9] need compute all potential SLCAs and then compute the probabilities for them to be SLCAs. And the real SLCAs will be picked out finally. But there are so many SLCA candidate nodes that extra work is needed to find the real SLCA nodes in [9].

Number of Query: Our algorithm only needs to scan and parse the XML document once regardless of dataset size. So a batch of queries can make full use of the algorithm ability. We randomly generate queries (query type is 2-100-100) and run them on datasets $Doc11$ and $Doc12$. Fig. 5 shows that the average query time slowly decreases and finally stabilizes as the number of query increases.
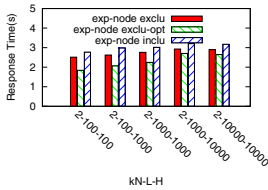
Fig. 6. Keyword Frequency's Effect
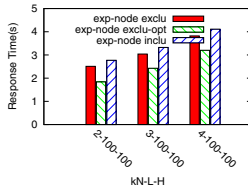
Fig. 7. #Keyword's Effect
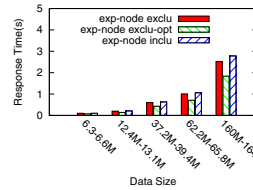
Fig. 8. Data Size's Effect

Frequency of Keywords: Fig. 6 records the average execution time of different patterns queries on $Doc11$ and $Doc12$. Fig. 6 shows the average execution time doesn't change significantly when the keyword frequency increases from 100 to 10,000. Therefore, algorithms aren't sensitive to the keyword frequency.

Number of Keywords: Fig. 7 shows the influence of the keyword number on the execution time. The number of keywords increases from 2 to 4. The queries

run on $Doc11$ and $Doc12$. Experimental results indicate that the execution time increases linearly when the number of keywords increases from 2 to 4.

Dataset size: Fig. 8 demonstrates the influence of dataset size on the average execution time. We randomly generate queries (type is 2-100-100) and run them on $Doc11$ and $Doc12$. Experimental result indicates the average execution time increases linearly when the dataset size increases.

6   Conclusions

This paper works on keyword filtering over probabilistic XML data model $PrXML^{\{exp,ind,mux\}}$. We analyze probability computation, define several operators concerning the computation and then give the whole algorithms which computes the SLCA probability by scanning the document only once bottom-up. Experimental results demonstrate the expansibility of our algorithms on data type, dataset size and query times. Our algorithm is insensitive to query keywords frequency and suitable for high-frequency keyword filtering.

References

1. Senellart, P., Abiteboul, S.: On the complexity of managing probabilistic xml data. In: PODS, pp. 283–292 (2007)
2. Nierman, A., Jagadish, H.V.: Protdb: Probabilistic data in xml. In: VLDB, pp. 646–657 (2002)
3. van Keulen, M., de Keijzer, A., Alink, W.: A probabilistic xml approach to data integration. In: ICDE, pp. 459–470 (2005)
4. Abiteboul, S., Senellart, P.: Querying and Updating Probabilistic Information in XML. In: Ioannidis, Y., Scholl, M.H., Schmidt, J.W., Matthes, F., Hatzopoulos, M., Böhm, K., Kemper, A., Grust, T., Böhm, C. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 1059–1068. Springer, Heidelberg (2006)
5. Abiteboul, S., Kimelfeld, B., Sagiv, Y., Senellart, P.: On the expressiveness of probabilistic xml models. VLDB J. 18(5), 1041–1064 (2009)
6. Hung, E., Getoor, L., Subrahmanian, V.S.: Probabilistic Interval XML. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) ICDT 2003. LNCS, vol. 2572, pp. 358–374. Springer, Heidelberg (2002)
7. Kimelfeld, B., Kosharovsky, Y., Sagiv, Y.: Query efficiency in probabilistic xml models. In: SIGMOD Conference, pp. 701–714 (2008)
8. Chang, L., Yu, J.X., Qin, L.: Query ranking in probabilistic xml data. In: EDBT, pp. 156–167 (2009)
9. Li, J., Liu, C., Zhou, R., Wang, W.: Top-k keyword search over probabilistic xml data. In: ICDE, pp. 673–684 (2011)
10. Xu, Y., Papakonstantinou, Y.: Efficient keyword search for smallest lcas in xml databases. In: SIGMOD Conference, pp. 537–538 (2005)
11. Sun, C., Chan, C.Y., Goenka, A.K.: Multiway slca-based keyword search in xml data. In: WWW, pp. 1043–1052 (2007)

H-Tree: A Hybrid Structure for Confidence Computation in Probabilistic Databases

Qian Zhang[1,2], Biao Qin[1,2], and Shan Wang[1,2]

[1] Key Laboratory of Data Engineering and Knowledge Engineering
(Renmin University of China), Ministry of Education, Beijing 100872, China
[2] School of Information, Renmin University of China, Beijing 100872, China
{zhangqian15,qinbiao,swang}@ruc.edu.cn

Abstract. Probabilistic database has become a popular tool for uncertain data management. Most work in the area is focused on efficient query processing and has two main directions, accurate or approximate evaluation. In recent work for conjunctive query without self-joins on a tuple-independent probabilistic database, query evaluation is equivalent to computing marginal probabilities of boolean formulas associated with query results. If formulas can be factorized into a read-once form where every variable appears at most once, confidence computation is reduced to a tractable problem that can be evaluated in linear time. Otherwise, it is regarded as a NP-hard problem and need to be evaluated approximately. In this paper, we propose a framework that evaluates both tractable and NP-hard conjunctive queries efficiently. First, we develop a novel structure H-tree, where boolean formulas are decomposed to small partitions which are either read-once or NP-hard. Then we propose algorithms for building H-tree and parallelizing (approximate) confidence computation. We also propose fundamental theorems to ensure the correctness of our approaches. Performance experiments demonstrate the benefits of H-tree, especially for approximate confidence evaluation on NP-hard queries.

Keywords: probabilistic database, conjunctive query, approximate confidence computation, read-once function.

1 Introduction

Uncertain data emerging in real-world applications has led to the renewed interests in probabilistic database. Most work of the area is concerned about uncertain data model and probabilistic query processing. There are two basic types of data uncertainty in probabilistic databases, *tuple-existence* and *attribute-value* uncertainty [1]. Various models have been proposed for expressing data uncertainty and correlations, and systems have been developed based on them for efficient query evaluation [1] [3] [4] [7] [11] [12] [16]. As an example, in a tuple-independent probabilistic database as Mystiq, each tuple is specified with an existence probability p, which means it belongs to the table with p and is absent with $1 - p$. Probabilistic database has become a popular tool for uncertain data management.

In a probabilistic database, query evaluation is #P-complete based on possible worlds [2]. Systems either evaluate queries of certain classes which allow for accurate confidence computation in linear time [2] [10] [11], or compute approximate results with controllable error [6] [7] [8] [9]. As an example, *safe/hierarchical queries* can be evaluated in PTIME on a tuple-independent probabilistic database, while [10] and [6] suggest that the class is likely to be too restrictive. In recent work for conjunctive query without self-joins on a tuple-independent probabilistic database, query evaluation is equivalent to computing marginal probabilities of boolean formulas associated with query results. Specifically, according to possible world semantics, a query result r is the disjunction of the identical r returned by all possible worlds, and its marginal probability is equal to the probability of the corresponding formula holding true. If the formula can be factorized into a read-once form where every variable appears at most once (e.g. $x \wedge (y \vee z)$), confidence computation is reduced to a tractable problem which can be evaluated in linear time [13] [14]. Otherwise, it is regarded as a NP-hard problem which cost increases exponentially. NP-hard queries can be evaluated approximately by formula factorization [8] [9] and Monte-Carlo simulation [5] [7].

In this paper, we study the problem of given a set of boolean formulas $\{\Phi_1 \ldots \Phi_n\}$ associated with query results $\{r_1 \ldots r_n\}$ of a conjunctive query without self-joins on a tuple-independent probabilistic database, computing accurate probabilities for read-once formulas and approximate probabilities for NP-hard ones. A naive approach is to evaluate each formula first with an algorithm which identifies read-once formulas and computes accurate probabilities. If the algorithm finds that the formula is NP-hard, an approximate algorithm will be invoked. Obviously, the naive approach may duplicate formula factorization and double the cost for NP-hard formulas.

There is one important observation that **boolean formulas, regardless of read-once or NP-hard, can be factorized into pairwise independent formulas.** While read-once formulas can be decomposed to smaller read-once partitions, NP-hard formulas can also be factorized into smaller read-once partitions and NP-hard partitions. For Example 1, while boolean formula Φ is NP-hard, it contains read-once partitions $x_4 y_5 z_2 \vee x_5 y_5 z_2$ and $x_8 y_8 z_3 \vee x_8 y_9 z_3$. By the observation, we propose a novel structure *H-tree* to represent boolean formulas regardless of read-once or not, and parallelize confidence computation based on that. That is, our approach integrates accurate and approximate confidence computation for probabilistic query processing, and parallelizes the evaluation on a much smaller scale of the original problem.

Our main contributions are concluded as follows:

- We propose a unified solution of evaluating boolean formulas associated with results of a conjunctive query without self-joins on a tuple-independent probabilistic database. It outputs accurate probabilities for read-once formulas and approximate probabilities with a fixed error for NP-hard formulas.
- We propose a novel structure *H-tree* which is able to express read-once and NP-hard formulas. Specifically, we compile each formula into an *H-tree*, which is composed of $\otimes$ and $\odot$ (representing $\vee$ and $\wedge$ between independent formulas) as internal nodes and *minimal read-once/NP-hard sub-formulas* as leaves (details in Section 3).

- Based on H-tree, we propose fundamental theorems and algorithms for parallel (approximate) evaluation on formulas.
- Experimental evaluation demonstrates that, for complex boolean formulas, our approach outperforms the state of the art which is designed for evaluating either read-once or NP-hard formulas.

Example 1. Fig. 1 shows a tuple-independent probabilistic database with relations $\underline{R}$, $\underline{S}$, $\underline{T}$ composed of $\{x_1 \ldots x_8\}$, $\{y_1 \ldots y_9\}$ and $\{z_1 \ldots z_4\}$ respectively. We evaluate conjunctive query Q on the database, which produces a boolean result with the corresponding formula Φ.

(R)	A	B		(S)	A	B	C		(T)	C	
x_1	1	3	p_{11}	y_1	1	1	1	p_{21}	z_1	1	p_{31}
x_2	1	2	p_{12}	y_2	1	2	1	p_{22}	z_2	2	p_{32}
x_3	2	3	p_{13}	y_3	2	1	1	p_{23}	z_3	3	p_{33}
x_4	3	2	p_{14}	y_4	2	2	1	p_{24}	z_4	1	p_{34}
x_5	3	2	p_{15}	y_5	3	1	2	p_{25}			
x_6	4	3	p_{16}	y_6	4	1	3	p_{26}			
x_7	4	2	p_{17}	y_7	4	2	3	p_{27}			
x_8	5	3	p_{18}	y_8	5	1	3	p_{28}			
				y_9	5	2	3	p_{29}			

Q: -$\underline{R}$ (A), $\underline{S}$ (A, B), $\underline{T}$ (C), $\underline{R}$.A = $\underline{S}$.A, $\underline{R}$.B > $\underline{S}$.B and $\underline{S}$.C = $\underline{T}$.C

$\Phi = x_1y_1z_1 \lor x_1y_1z_4 \lor x_1y_2z_1 \lor x_1y_2z_4 \lor x_2y_1z_1 \lor x_2y_1z_4$
$\lor x_3y_3z_1 \lor x_3y_3z_4 \lor x_3y_4z_1 \lor x_3y_4z_4 \lor \underline{x_4y_5z_2} \lor \underline{x_5y_5z_2}$
$\lor x_6y_6z_3 \lor x_6y_7z_3 \lor x_7y_6z_3 \lor \underline{x_8y_8z_3} \lor \underline{x_8y_9z_3}$

Fig. 1. A tuple-independent probabilistic database $\{\underline{R},\ \underline{S},\ \underline{T}\}$ and conjunctive query Q with the corresponding formula Φ

In the next section, we provide background knowledge for the concepts used in this paper. In Section 3, we propose an algorithm to build *H-tree* for a boolean formula. In Section 4, we provide techniques to parallelize accurate/approximate confidence evaluation based on *H-tree*. Experimental results demonstrate the performance of our approach in Section 5. We conclude the paper in Section 6.

2   Preliminary

In this section, we present background knowledge of conjunctive query without self-joins on a tuple-independent probabilistic database and read-once formula.

2.1   Conjunctive Query without Self-joins on Tuple-Independent Probabilistic Database

In a tuple-independent probabilistic database, each tuple t_i is present in the table with an existence probability p_i and absent with $1-p_i$. The existence of tuples is independent from each other. We associate t_i with a unique binary random variable x_i such that $x_i =$

1 if t_i is present in the table, otherwise $x_i = 0$. That is, $p_i = \Pr(x_i = 1)$. When it is clear from the context, we refer to the tuple by its random variable as in Example 1. A tuple-independent probabilistic database $D = \{R_1, \ldots, R_m\}$ is a set of relations composed of such tuples, and represents a distribution over possible worlds each obtained by choosing a (sub)set of tuples in R_i to be present.

A conjunctive query corresponds to a *select-project-join* query in SQL, with (in)equi-joins and conjunctions in *where* clause [15]. Generally, given conjunctive query q on probabilistic database D, query results are defined to be the union of results returned by each possible world, and the marginal probability of each result r is the sum of probabilities of possible worlds that return r. Prior work [1] [13] [16] has illustrated the equivalence between query evaluation under possible world semantics and via boolean formulas. During the evaluation, a boolean formula is built for each result, and the result's probability is equal to the probability of the formula holding true. Let the boolean formula for result r be Φ_r, the extended definitions of relational algebra operators σ, $\times$, and Π for building formulas are:

$$\Phi_t = x_t, \forall t \in R$$
$$\Phi_{\sigma_c(t)} = \text{if } c(t) \text{ then } \Phi_t \text{ else F}$$
$$\Phi_{t \times t'} = \Phi_t \wedge \Phi_{t'}$$
$$\Phi_{\Pi(t_1 \ldots t_k)} = \vee_{i=1}^k \Phi_{t_i}$$

Approximate confidence computation with a fixed error is defined as follows, and we focus on the absolute ε-approximation in this paper.

Definition 1 (ε-approximation). A value *p'* is an absolute (or additive) ε-approximation of a probability *p* if $p-\varepsilon \leq p' \leq p+\varepsilon$. A value *p'* is a relative (or multiplicative) ε-approximation of a probability *p* if $(1-\varepsilon)*p \leq p' \leq (1+\varepsilon)*p$. [8]

2.2   Read-Once Formula

The problem of conjunctive query evaluation on probabilistic database is #P-complete. However, if the corresponding formula can be factorized into a *read-once* form where every variable appears at most once, the accurate probability can be computed in polynomial time. Given a boolean formula Φ, a conjunction of variables is called a clause. A formula in disjunctive normal form (DNF) is a disjunction of clauses. A prime clause of Φ is a clause with a minimal set of variables among all that can appear in DNFs equivalent to Φ [14].

Theorem 1. A boolean formula Φ is read-once iff it is unate, P_4-free and normal. [17]

Φ is unate if every variable appears in either positive or negative form. The co-occurrence graph G_{CO} for Φ is an undirected graph with the vertex set $V(G_{CO})$ equal to the variable set of Φ. The edge set $E(G_{CO})$ is defined as follows: there is an edge between two variables iff they appear in the same prime clause of Φ. G_{CO} is uniquely determined by Φ. Fig. 2 (a) shows an example of the co-occurrence graph. The special graph P_4 denotes a chordless path with 4 vertices and 3 edges (Fig. 2 (b)). Let X

denote a subset of vertices of G_{CO}, the subgraph of G_{CO} induced by X is formed by restricting edges of G_{CO} to edges with end points in X. Φ is P_4-free if no induced subgraph of its G_{CO} forms a P_4. Φ is normal if every clique in its G_{CO} is contained in some clause of its DNF.
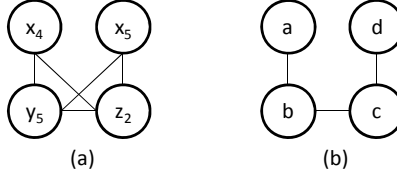
Fig. 2. (a) G_{CO} for the read-once formula $x_4 y_5 z_2 \vee x_5 y_5 z_2$ in Example 1 (b) P_4 graph

Theorem 2. Let Φ be a boolean formula associated with a result tuple produced by a conjunctive query without self-joins on a tuple-independent probabilistic database. Φ is a unate, normal, and k-monotone DNF (k is the number of relations involved). [13]

3   Formula Factorization

In this section, we propose a novel structure called *H-tree* for compiling boolean formulas produced by a conjunctive query without self-joins on a tuple-independent probabilistic database. We present several definitions for *H-tree*, and provide an algorithm that compiles a boolean formula to a complete *H-tree* regardless of read-once or not. We also discuss the relationship between our algorithm and the state of the art.

3.1   Definition of H-Tree

Given boolean formulas Φ and Φ_1, we denote $\Phi_1 \subseteq \Phi$ if Φ_1 is neither null nor a constant (true or false), and variables of Φ_1 all appear in Φ. Let $\otimes$ represents for $\bigvee$ (logical "or") and $\odot$ for $\bigwedge$ (logical "and") between independent formulas, we apply two decomposition methods in this paper:

- Independent-or: Factorize Φ into independent $\Phi_1, \Phi_2 \subset \Phi$ s.t. $\Phi = \Phi_1 \otimes \Phi_2$;
- Independent-and: Factorize Φ into independent $\Phi_1, \Phi_2 \subset \Phi$ s.t. $\Phi = \Phi_1 \odot \Phi_2$.

Definition 2 (Sub-formula). Given a DNF Φ, Φ_1 is a *sub-formula* of Φ if $\Phi_1 \subseteq \Phi$, and there exists a boolean formula Φ_2 s.t. $\Phi = \Phi_1 \otimes \Phi_2$ or $\Phi = \Phi_1 \odot \Phi_2$. If $\Phi_2 \subseteq \Phi$, Φ_2 is called $\otimes$-*complement* or $\odot$-*complement* of Φ_1 on Φ, respectively.
Obviously, for any formula Φ, it is a sub-formula for itself.

Definition 3 (Naive Formula). Given a DNF Φ, it is called *naive* if it contains less than 3 clauses, or its co-occurrence graph G_{CO} contains less than 4 vertices or less than 3 edges. Furthermore, if Φ is read-once, it is called *naive read-once*.

A *naive* formula is P_4-free. However, it may be not read-once in some cases. For example $\Phi = xy \lor yz \lor zx$, Φ is not read-once for it is not normal.

Definition 4 (Minimal Read-once/NP-hard Sub-formula). Given a boolean formula Φ, Φ_1 is a *minimal read-once* sub-formula of Φ if Φ_1 is *naive read-once* and a sub-formula of Φ. Meanwhile, Φ_2 is a *minimal NP-hard* sub-formula of Φ if Φ_2 is a NP-hard sub-formula of Φ and has no sub-formula except Φ_2 itself.

Definition 5 (H-tree). Given a boolean formula Φ, a complete *H-tree* for Φ is a tree structure with $\otimes$ and $\odot$ as internal nodes and *minimal read-once/NP-hard sub-formulas* of Φ as leaves. Given a *partial* H-tree, the H-tree obtained by replacing a leaf by an equivalent (partial) H-tree is called a *refinement*.
Fig. 3 shows the complete H-tree for formula Φ of Example 1.

Theorem 3. Given a boolean formula Φ, there must exist a complete H-tree for Φ.

Proof. We present the following steps for building a complete H-tree M_t for Φ:

- If Φ is read-once, we can compile it to a complete H-tree where all leaves are *minimal read-once sub-formulas* of Φ by state-of-the-art read-once algorithms, such as [14] [17].
- If not, let M_t be a (partial) H-tree with Φ as its only node. If Φ is a *minimal NP-hard sub-formula* for itself, M_t is a complete *H-tree* for Φ.
- If not, let Φ_1, Φ_2 be sub-formulas of Φ s.t. $\Phi = \Phi_1 \otimes \Phi_2$ or $\Phi = \Phi_1 \odot \Phi_2$, refine M_t by replacing Φ with Φ_1, Φ_2 and the corresponding operator.
- Restart to handle Φ_1 and Φ_2 as Φ from the first step, respectively.

Finally, all internal nodes of M_t are $\otimes$ and $\odot$, and all leaves are *minimal read-once/NP-hard sub-formulas* of Φ. M_t is a complete H-tree for Φ.   ∎

3.2   H-Tree Generation

Based on Theorem 3, given a boolean formula Φ produced by a conjunctive query without self-joins on a tuple-independent probabilistic database, we propose Algorithm 1 that compiles Φ to a complete H-tree. We demonstrate Algorithm 1 by compiling formula Φ of Example 1, and the H-tree is shown in Fig. 3.

Theorem 4. For a boolean formula Φ produced by a conjunctive query without self-joins on a tuple-independent probabilistic database, Algorithm 1 builds a complete H-tree M_t for Φ.

Proof. First, for a *naive* leaf of M_t built by Algorithm 1 is unite, normal and P_4-free by Definition 3 and Theorem 2, it is obviously *naive read-once* by Theorem 1.

Then similar to Theorem 3, we can prove that the algorithm covers all situations that could happen.

For all internal nodes of M_t built by Algorithm 1 are $\otimes$ and $\odot$, and all leaves of M_t are either *minimal read-once* or *minimal NP-hard sub-formulas* of Φ, M_t is a complete H-tree for Φ.   ∎

Algorithm 1. *Compile(Φ)*

Input: Φ, a boolean formula produced by a conjunctive query without self-joins on a tuple-independent probabilistic database

Output: M_t, a complete H-tree for Φ

1. Make an H-tree M_t with Φ as its only node;
2. If Φ is *naive* then
3.   Return M_t;
4. End if
5. If Φ is *minimal NP-hard* then
6.   Return M_t;
7. End if
8. Compute sub-formulas Φ_1, Φ_2 of Φ s.t. $\Phi = \Phi_1 \otimes \Phi_2$ or $\Phi = \Phi_1 \odot \Phi_2$;
9. Refine M_t by replacing Φ with Φ_1, Φ_2 and the corresponding operator;
10. Compile(Φ_1);
11. Compile(Φ_2);
12. Return M_t;
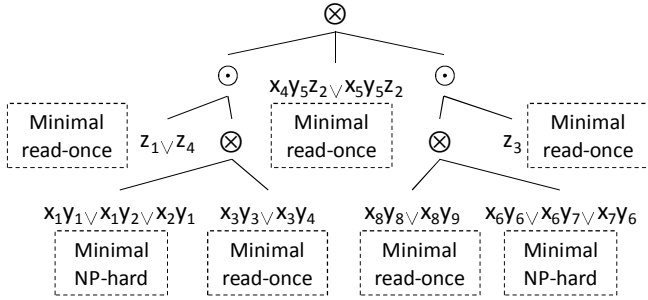
Fig. 3. A complete H-tree built by Algorithm 1 for formula Φ of Example 1

When computing sub-formulas Φ_1, Φ_2 of Φ, we can utilize state-of-the-art read-once algorithms such as [14] [17]. In fact, they compile Φ to $\Phi_1 \otimes \Phi_2$ or $\Phi_1 \odot \Phi_2$, which just satisfy the needs of our algorithm. The relationship between Algorithm 1 and the state of the art is as follows:

- The purpose for Algorithm 1 is to factorize a boolean formula, regardless of read-once or NP-hard, into small partitions. While Algorithm 1 utilizes read-once algorithms to compute sub-formulas, it handles NP-hard formulas seamlessly.
- Unlike the state of the art that decomposes the boolean formula until clauses or variables, we define two kinds of formulas as leaves of a complete H-tree, which makes the compilation stop at a proper level for parallelizing the confidence computation in the next section.

Let *n'* be the number of variables involved in the input formula at the current recursive call (with different occurrences of the same variable counted multiple times), we analysis the time complexity of Algorithm 1 as follows:

- The process checking a *naive* formula takes O(n') at the current recursive call.
- Using state-of-the-art read-once algorithms, Step 5 and 8 can be evaluated at the same time. Specifically, when the read-once algorithm triggers its NP-hard conditions, the input formula at the current recursive call is actually *minimal NP-hard*.
- For we have defined two specific leaves for a complete H-tree, while the time complexity of Algorithm 1 is similar to the read-once algorithm's, our algorithm has less recursive calls and a lower tree structure.

4   Confidence Computation

For read-once formulas, Algorithm 1 builds complete H-trees with only *naive read-once* leaves. For NP-hard formulas, it builds H-trees with *minimal NP-hard* leaves and maybe with *naive read-once* leaves as well. All leaves of an H-tree are pairwise independent. Based on that, we propose a parallelized confidence computation algorithm in this section.

As an example, we use the algorithm [14] for accurate confidence computation of read-once leaves in an H-tree, and the algorithm [8] for approximate confidence computation of NP-hard leaves. Due to space limitations, we do not detail these algorithms here. Generally, [14] computes accurate probabilities for read-once formulas, and [8] computes the lower and upper probability bounds of formulas with ε-approximation. For formulas with the decomposition types ⊗ and ⊙ are increasing monotonically, the probability bounds can be propagated from leaves to the root of an H-tree. With the complete H-tree M_t built by Algorithm 1, we present a parallel algorithm for the ε-approximate confidence computation (Algorithm 2). The time complexity of Algorithm 2 is determined by the largest *minimal NP-hard* leaf of M_t, and is equal to [8] when the original formula is *minimal NP-hard* itself.

Algorithm 2. *Para_Appro(M_t, ε)*

Input: M_t, a complete H-tree with m NP-hard leaves
        ε, a parameter for approximate confidence computation
Output: [L, U], the lower and upper probability bounds of M_t
1.   Foreach leaf Φ in M_t
2.   If Φ is *naive read-once* then
3.   Create a thread to compute p as the algorithm [14];
4.   Else
5.   Create a thread to compute [l, u] satisfying **ε/m** as the algorithm [8];
6.   End if
7.   End foreach
8.   Compute [L, U] for M_t from bottom to up;
9.   Return [L, U];

Theorem 5. Given a complete H-tree M_t produced by Algorithm 1 for a NP-hard formula Φ and ε for its approximate confidence computation, Algorithm 2 generates the probability bounds [L, U] that satisfy ε for Φ.

Proof. Given an internal node r of M_t, let l be the number of r's children $\{r_i \mid P(r_i) = p_i \in [a_i, b_i], a_i = p_i - \varepsilon_i, b_i = p_i + \varepsilon_i\}$. The probability bounds $[a', b']$ of r can be computed as follows (let $a' = p' - \varepsilon'$, $b' = p' + \varepsilon'$).

Case 1: If r is $\otimes$, we have:

$$a' = p' - \varepsilon' = 1 - \Pi(1 - p_i + \varepsilon_i)$$
$$b' = p' + \varepsilon' = 1 - \Pi(1 - p_i - \varepsilon_i)$$
$$\varepsilon' = (\Pi(1 - p_i + \varepsilon_i) - \Pi(1 - p_i - \varepsilon_i))/2$$

In the following, we prove that $\varepsilon' \leq \sum \varepsilon_i$ by induction on l. When $l = 2$, we have $\varepsilon' = \varepsilon_2(1 - p_1) + \varepsilon_1(1 - p_2) < \varepsilon_1 + \varepsilon_2$. Suppose that the induction hypothesis holds with $l - 1$ and consider l ($l > 2$), we have:

$$\varepsilon' = (\textstyle\prod_{l-1}(1 - p_i + \varepsilon_i)(1 - p_l + \varepsilon_l) - \prod_{l-1}(1 - p_i - \varepsilon_i)(1 - p_l - \varepsilon_l))/2$$
$$= (1 - p_l)(\textstyle\prod_{l-1}(1 - p_i + \varepsilon_i) - \prod_{l-1}(1 - p_i - \varepsilon_i))/2$$
$$+ \varepsilon_l(\textstyle\prod_{l-1}(1 - p_i + \varepsilon_i) + \prod_{l-1}(1 - p_i - \varepsilon_i)/2$$
$$\varepsilon' < (1 - p_l)\textstyle\sum_{l-1}\varepsilon_i + \varepsilon_l < \sum\varepsilon_i$$

Case 2: If r is $\odot$, we have:

$$a' = p' - \varepsilon' = \Pi(p_i - \varepsilon_i)$$
$$b' = p' + \varepsilon' = \Pi(p_i + \varepsilon_i)$$
$$\varepsilon' = (\Pi(p_i + \varepsilon_i) - \Pi(p_i - \varepsilon_i))/2$$

As Case 1, we prove $\varepsilon' \leq \sum\varepsilon_i$ by induction on l similarly.

For read-once leaves have $\varepsilon = 0$, the probability bounds [L, U] generated by Algorithm 2 satisfy ε for Φ. ∎

Obviously, the algorithms [14] and [8] used in Algorithm 2 can be replaced by other state-of-the-art algorithms that evaluate read-once and NP-hard formulas respectively. Specially, while Algorithm 1 and 2 are seemingly similar to [8] with approximate confidence computation for NP-hard formulas, Algorithm 1 factorizes the input formula into *minimal read-once/NP-hard* sub-formulas as a complete H-tree, and Algorithm 2 makes the approximation be processed in parallel and only on the NP-hard sub-formulas.

5 Performance Evaluation

This section presents performance experiments for our approach (called H-tree). The experiments are performed on a workstation with 5G memory, 160G disk and four cores CPU of 3.6GHz respectively. We build the prototype as a middleware on an underlying database *PostgresSQL* 8.3 by Java 1.5 on Ubuntu 9.10 operating system. Experimental data is generated by a data generator which creates tuple-independent probabilistic databases with various scale factors, where scale factor = 1 indicates that there are one million result tuples involved in the processing. For boolean formulas

are either read-once or NP-hard, we specify five kinds of formulas shown in Table 1 for the experiments. Experimental results demonstrate that, for complex boolean formulas, H-tree outperforms the algorithms [14] (called CompRO) and [8] (called D-tree) which are designed for evaluating read-once and NP-hard formulas respectively. The efficiency of H-tree is impressive especially when evaluating complex NP-hard formulas with a small ε.

Table 1. Five kinds of boolean formulas used in experiments

Formula kinds	Features
Φ_1	Naive read-once
Φ_2	Read-once
Φ_3	Minimal NP-hard
Φ_4	Composed of one NP-hard and several read-once sub-formulas
Φ_5	Composed of several NP-hard sub-formulas

5.1   Read-Once Evaluation

In this section, we compute accurate probabilities for read-once formulas and compare the efficiency of H-tree to the algorithm [14] (called CompRO in the sequel). Experimental results are shown in Fig. 4, where "*n* naive" indicates that the input formula can be factorized into *n minimal read-once* sub-formulas. Specifically, CompRO computes the probability along with formula compilation, while our approach compiles the formula to H-tree first and computes the probability in parallel based on it. That is, our approach needs to traverse the tree twice. When the input formula is *naive read-once*, there is no gain in parallelization, and the execution time of H-tree is almost twice that of CompRO. However, when the input formula is composed of 3 or 4 *minimal read-once* sub-formulas, H-tree is more efficient than CompRO as shown in Fig. 4. Compared to CompRO, our approach performs better when the read-once formula is larger.
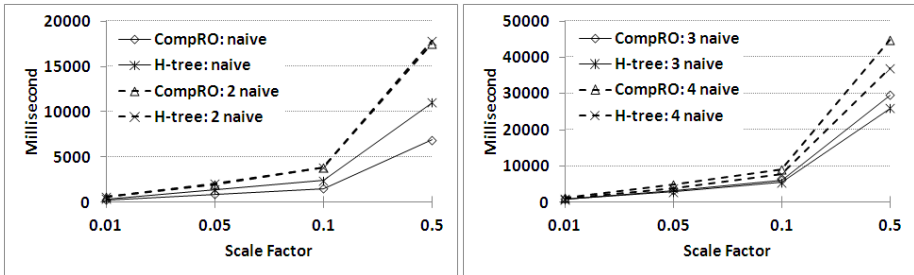

Fig. 4. Read-once formula evaluation

5.2   NP-Hard Evaluation

In this section, we compute approximate probabilities for NP-hard formulas and compare the efficiency of H-tree to the algorithm [8] (called D-tree in the sequel).

Experimental results are shown in Fig. 5. For the same reason with *naive read-once* formulas, the execution time of H-tree is about 1.25 times of D-tree's with *minimal NP-hard* formulas, as shown in Fig. 5 (a). When the input formula can be decomposed to several sub-formulas, the execution time of H-tree is determined by the largest *minimal NP-hard* sub-formula. As shown in Fig. 5 (b), when the input formula is composed of one *minimal NP-hard* sub-formula and several read-once sub-formulas, H-tree just needs to approximate the *NP-hard* sub-formula and is more efficient than D-tree. In Fig. 5 (c), when the input formula can be decomposed to two *minimal NP-hard* sub-formulas, the execution time of H-tree is almost half of D-tree's. On the other hand, for the probability of the NP-hard formula is approximated along with formula factorization, one decomposition operation may get tighter probability bounds for the formula that satisfy several values of ε, as shown in Fig. 5 (d). It may help users to choose a proper ε when evaluating NP-hard formulas.

(a) Φ_3 evaluation

(b) Φ_4 evaluation

(c) Φ_5 evaluation

(d) H-tree evaluation with scale factor 0.1
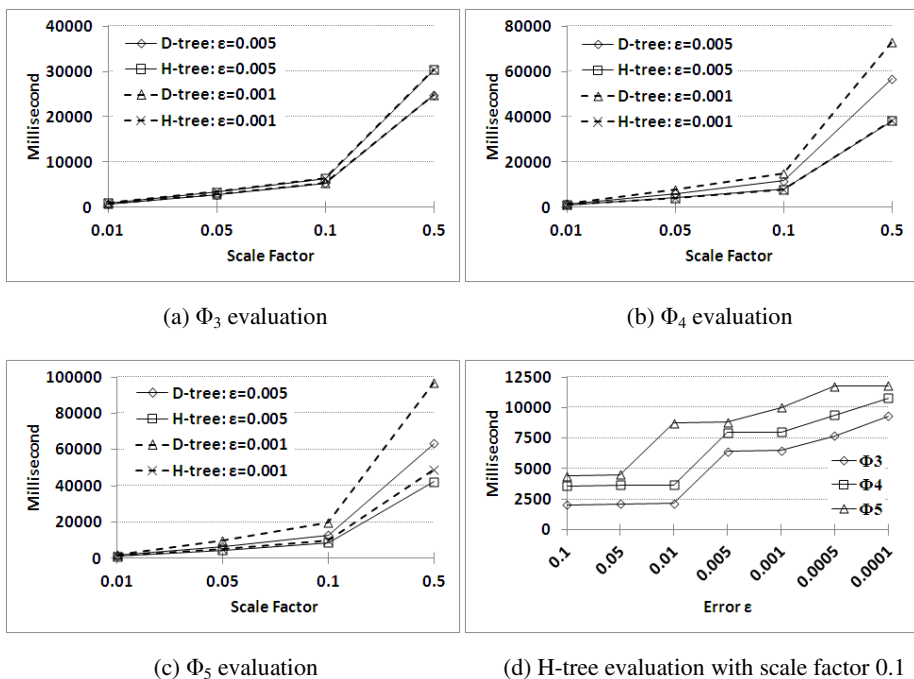
Fig. 5. NP-hard formula evaluation

6 Conclusion

In this paper, we propose a framework for evaluation on both tractable and NP-hard conjunctive queries without self-joins on a tuple-independent probabilistic database. We develop a novel structure H-tree, where boolean formulas can be factorized into small read-once or NP-hard partitions. We parallelize confidence computation for query results based on it, and provide accurate probabilities for read-once formulas and approximate probabilities with a fixed error for NP-hard ones. Experimental

results demonstrate that our approach outperforms the state of the art with complex boolean formulas, especially for NP-hard ones.

References

1. Benjelloun, O., Sarma, A.D., Halevy, A., Widom, J.: ULDBs: databases with uncertainty and lineage. In: VLDB 2006, Seoul, Korea (2006)
2. Dalvi, N., Suciu, D.: Efficient Query Evaluation on Probabilistic Databases. In: VLDB 2004, Toronto, Canada (2004)
3. Antova, L., Jansen, T., Koch, C., Olteanu, D.: Fast and Simple Relational Processing of Uncertain Data. In: ICDE 2008, Cancún, México (2008)
4. Boulos, J., Dalvi, N., Mandhani, B., Mathur, S., Re, C., Suciu, D.: MYSTIQ: a system for finding more answers by using probabilities. In: SIGMOD 2005, Baltimore, Maryland, USA (2005)
5. Ré, C., Dalvi, N., Suciu, D.: Efficient top-k query evaluation on probabilistic data. In: ICDE 2007, Istanbul, Turkey (2007)
6. Koch, C.: Approximating predicates and expressive queries on probabilistic databases. In: PODS 2008, Vancouver, BC, Canada (2008)
7. Jampani, R., Xu, F., Wu, M., Perez, L.L., Jermaine, C.M., Haas, P.J.: MCDB: A monte carlo approach to managing uncertain data. In: SIGMOD 2008, Vancouver, BC, Canada (2008)
8. Olteanu, D., Huang, J., Koch, C.: Approximate confidence computation in probabilistic databases. In: ICDE 2010, Long Beach, California, USA (2010)
9. Fink, R., Olteanu, D.: On the optimal approximation of queries using tractable propositional languages. In: ICDT 2011, Uppsala, Sweden (2011)
10. Dalvi, N., Suciu, D.: The dichotomy of conjunctive queries on probabilistic structures. In: PODS 2007, Beijing, China (2007)
11. Ré, C., Suciu, D.: Materialized views in probabilistic databases: for information exchange and query optimization. In: VLDB 2007, Vienna, Austria (2007)
12. Olteanu, D., Huang, J., Koch, C.: SPROUT: Lazy vs. eager query plans for tuple-independent probabilistic databases. In: ICDE 2009, Shanghai, China (2009)
13. Sen, P., Deshpande, A., Getoor, L.: Read-once functions and query evaluation in probabilistic databases. In: Proceedings of the VLDB Endowment 3(1-2), 1068–1079 (2010)
14. Roy, S., Perduca, V., Tannen, V.: Faster query answering in probabilistic databases using read-once functions. In: ICDT 2011, Uppsala, Sweden (2011)
15. Kanagal, B., Li, J., Deshpande, A.: Sensitivity analysis and explanations for rubost query evaluation in probabilistic databases. In: SIGMOD 2011, Athens, Greece (2011)
16. Sen, P., Deshpande, A., Getoor, L.: Prdb: managing and exploiting rich correlations in probabilistic databases. Proceedings of the VLDB Endowment 18(5), 1065–1090 (2009)
17. Golumbic, M.C., Mintz, A., Rotics, U.: Factoring and recognition of read-once functions using cographs and normality and the readability of functions associated with partial k-trees. Discrete Applied Mathematics 154(10), 1465–1477 (2006)

Predicting Online Auction Final Prices Using Time Series Splitting and Clustering

Takuya Yokotani, Hung-Hsuan Huang, and Kyoji Kawagoe

Ritsumeikan University,
Kusatsu-city, Shiga, Japan
yokotani@coms.ritsumei.ac.jp, huang@fc.ritsumei.ac.jp,
kawagoe@is.ritsumei.ac.jp

Abstract. Online auctions allows users to sell and buy a variety of goods, and they are now one of the most important web services. Predicting final prices on online auctions is a hard task. However, there has been much pioneering work over the past ten years. In this paper, we propose a novel method for predicting final prices using a time series splitting and clustering method to provide higher accuracy. An evaluation of the effectiveness of our method is also described in the paper.

Keywords: Time series, Prediction, Online auction, Clustering.

1 Introduction

Online auctions allows users to sell and buy a variety of items and they have become one of the most important web services. Incresed activity on online auction web services has led to greater research work on the online auctions. Online auction final price prediction would be desirable for users, because it would allow them to win an auction item at the best price. There has been a lot of research work on final price prediction. Previous research can be separated into two main types from the viewpoint of input sources in final price prediction: 1) prediction using muti-attributes and 2) prediction using time series data.

Prediction using multi-attributes [1–4]. Final auction price dynamics can depend on multiple factors. It is common for the final price in an auction to be quite variable for the same product. Many factors can affect to the final auction price. These factors can include auction parameters such as auction length and free shipping. Item characteristics such as the seller ratting and "used or new" condition, are also important factors. Final price prediction classifiers have been developed for auction category characteristics [2]. In addition, some learning methods such as neural network and K-nearest neighbors have also been used to learn a model from many auction histories, which are represented as a sequence of multi attribute values [3].

Prediction using time series data [5–11]. The auction history of an item can be viewed as a time series data, i.e. a series of auction prices during an auction period. Several prediction methods have been proposed in recent

years. General and traditional prediction or forecasting methods include Box-Jenkins and ARIMA [14]. Other methods have also benn proposed, including fitting with special complex expressions and use of estimated statistical distributions [5, 10, 11]. The details of these models are described in [13].

In this paper, we present a novel prediction method for predicting final auction prices that belongs to the second class (Prediction using time series). It is assumed that only past auction histories are used for prediction of the final price of the current auction and the past auction histories used for a final price prediction of the current auction are selected before the final price prediction process. The reason for the first assumption is that we want to focus on the prediction accuracy based on price time series. However, the use of additional information such as auction parameters and item characteristics, such as Seller Rating, Last-minute Bidding and Item Condition, can contribute to a more accurate model. It is beneficial to produce higher accuracy by introducing additional information, in an time series prediction model. In the model presented in this paper, prediction is performed using only a subset of the vast amount of past auction histories. The time required for prediction can be greatly reduced by this pre-selection step. A subset of the auction histories can be extracted based on some predefined conditions related to the product type or the item characteristics. It is assumed that the number of auction histories used for prediction is determined by a user or a system manager. We do not discuss here an extraction method of the auction histories. Our method is based on splitting and clustering past auction histories. Before clustering commences, splitting is conducted to ensure high auction accuracy even when the small number of past histories is available for prediction.

The paper is organized as follows. First, we describe our motivation for the splitting and clustering method with a brief introduction. We then propose our novel final auction price prediction method, where we present a splitting and clustering based model in a formal manner, This is followed by an evaluation of the method. We conclude by discussing possible extensions to the model in the final section.

2   Motivation and Basic Ideas

2.1   Motivation

Figure 1, we consider an auction time series from the starting time until the current time. An auction user is eager to know the final auction price at the closing time. However, it is difficult to predict this because the future auction price is uncertain and it can increase depending on specific circumstances. If the price increases sharply, then the future price is very likely to be high, i.e., X. If there are no more bids in the auction, the price will be the same as the current price, i.e., Z. If the price increases steadily, the price will be the value Y. Therefore, it is difficult to predict the final price unless additional information is available. To more accurately estimate the possible future price in an auction, it is useful to refer to past auction histories of the same product. Figure 2 shows some
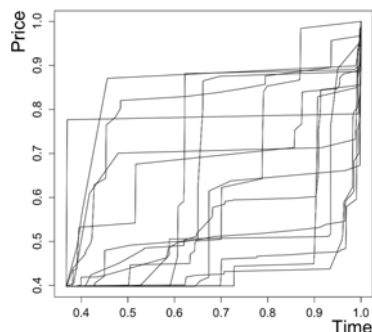
Fig. 1. Auction price dynamics

Fig. 2. Some auction price curves

past auction histories. Note that the price curves are based on the past auction histories of the same product on an actual auction site. Each auction price curve shown has been normalized for both price and time. As shown in the figure, there are many dynamic variations in auction prices. Thus, it is possible that a curve fitting method would produce inaccurate estimation of the auction final price. In this paper, we propose a novel method for predicting final prices using a time series splitting and clustering method which produces highly accurate predictions.

2.2   Basic Concepts

The basic concepts of our proposed prediction method are described here.

Clustering. The role of clustering in the prediction of final prices is shown in Fig.3. We assumed eight auction histories, from A to H in the figure. Given a query in a the current auction, Q, we can calculate the average of the final prices of A to H. Or we can obtain the final price of the best matched auction history from A to H, both of which may be different from the actual final price of Q. Models such as Grey system theory [5] and the functional data analysis [4], incorrect parameter identification can lead to many errors and icorect predictions of the final price. Auction histories can be divided into groups with similar curves using a clustering method and the prediction for the auction query Q can be applied to a group containing Q. For example, A and B found in Group G1, C and D are in Group G2, E and F are in Group G3, and G and H are in Group G4. If Q is included in Group G3, the final price of Q is estimated based only on the auction histories of G3 and by calculating the average final price of the past auctions in G3.

Splitting and normalization. A second problem when predicting a final auction price is that the current auction is underway whereas the past auction histories have already finished. Thus, the matching of the current auction curve and a past auction history curve should not be conducted only over the time interval from the starting time until the current time. Figure 4 shows
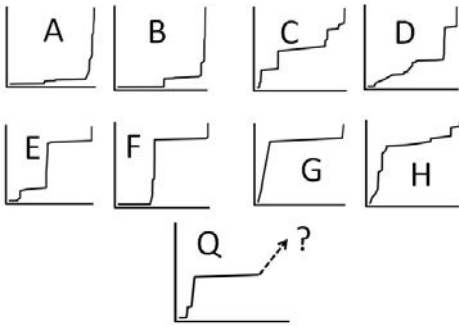
Fig. 3. Clustering of auction histories   **Fig. 4.** Matching with auction histories
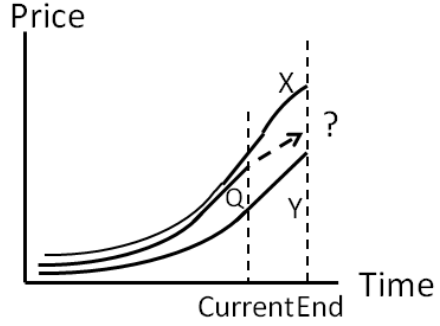
two past auction histories curves, X and Y, and the query auction curve Q. Note that the current time of Q is before the closing time of X and Y. In the figure, Q is very similar to X over the interval from the starting time until the current time. Y looks less similar to Q over the interval. If the matching of Q with past auction histories is only conducted over the interval, the predicted final price may be different from the actual final price of Q. To reduce this type of errors, we also consider the curve for Y as well as X when predicting the final price of Q. Note that the curves of Q and Y are more similar after Q is enlarged in the time dimension. Given the bidding situation in the actual auctions, we should not match the query curve with past curves only for the interval from the stating time to the current time. Thus, as shown for Y in Fig.4, we introduce time normalization and the splitting of auction histories.

3   Prediction Using Time Series Splitting and Clustering

In this section, we formally describe our proposed method. Table 1 defines the notations usded in this paper.

3.1   General Process

Given a time series $qdata$, we assume that we have a set of auction histories A containing past auction items for the same product. In our proposed method, the general prediction process is conducted as follows:

Step1: Splitting. We extract a set of split time series A' from A.

Step2: Normalization. We produce normalized time series D, which is obtained from the split time series A'. $qdata$ is also transformed into a normalized representation d_q.

Step3: Clustering. Both $D = \{d_k\}$ and d_q) are combined then segregated into NC clusters using a clustering method.

Step4: Prediction of the final price. The average final price is calculated for a set of the same cluster as d_q. If the number of elements in the cluster including d_q, is one, the average final price of A is calculated as the predicted price.

Table 1. Symbol definition

Symbol	Definition
N	the number of the auction time series
A	a set of the auction time series data, $A = \{a_i|i = 1, ..., N\}$
P_i	the final price of a_i
NB_i	the number of bidding in a_i
$T_i(j)$	bidding time in a_i, for $j = 1, .., NB_i$ and $T_i(1) = 0$
$p_i(j)$	bidding price at a time $T_i(j)$ in a_i, $P_i = p_i(NB_i)$
a_i	an auction time series of A described as a sequence of $< (T_i(j), p_i(j)) >$
NI	the number of split points including the closing time
S	a set of split time points, $S = \{S_k \in (0, 100]|k = 1, .., NI\}$,
S_k	a split time point in percentage from a start time until closing
I	a set of split intervals $I = \{I_k\}$
I_k	a split interval represented by $(1, End_i(k))$ for each a_i, $End_i(k) \in \{1, ..., NB_i\}$. $End_i(k) < End_i(k + 1)$
A'	set of split auction time series obtained from A. $A' = \{a'_k|k = 1, .., N * NI\}$
D'	set of price normalized split time series of A'. $D' = \{d'_k|k = 1, .., N * NI\}$
D	set of normalized split auction time series of A. $D = \{d_k|k = 1, .., N * NI\}$
NT	the number of time points in $[0.0, 1.0]$. $NT - 1$ is the no. of intervals. The number of intervals is $NT - 1$
δ	fixed interval length of $[0.0, 1.0]$ used for time normalization
T	a set of time points $T = \{t_s|s = 1, ..., NT\}$, $t_1 = 0.0$, $t_{NT} = 1.0$, $t_s = (s - 1) * \delta$
$f(,)$	$f(t_s, d'_k)$ is an interpolation function to calculate a value at t_s from d'_k where d'_k is an auction time series data after splitting.
d_k	an auction time series data after splitting and normalization, $d_k = < d_k(1), d_k(2), ..d_k(NT) > d_k \in D$ and $d_k(m) = f(t_m, d'_k)$
NC	the number of clusters
C	a set of clusters, $C = \{c_m|m_1, .., NC\}$
$qdata$	an auction time series data to predict its the final price $qdata$ is represented by a sequence of pairs $(T_q(j), q(i))$
NQ	the number of bidding in $qdata$ done by now
d_q	the normalized time series data of $qdata$

3.2   Time Series Splitting

The first step in predicting a final price is to split the auction time series into the same categories of query time series data $qdata$ that we need to predict. Note that the query time series $qdata$ dows not need to be split.

Assume that the number of biddings in a time series a_i to be split is NB_i and it is split into NI sub sequences. To do this, we split the integer intervals $[1, NB_i]$ into a set of integer intervals I, NI integer intervals I_k. $I_k = [1, End_i(k))]$, where

$End_i(k) = \lceil S_k/100 * NB_i \rceil$. Note that the starting value for each interval is 1. Then, the split time series a'_k of the original time series a_i is a sequence of pairs $(T_i(j), p_i(j)), j \in I_k$.

For example, suppose we have a time series $a = < (0, 4), (3, 30), (5, 34), (6, 40), (7, 60) >$, $NI = 2$ and $S = \{60, 100\}$. Then we obtain the following two split time series a'_1 and a'_2 because $END(1) = 3$ and $END(2) = 5$: $a'_1 = (0, 4), (3, 30), (5, 34) >$ and $a'_2 = < (0, 4), (3, 30), (5, 34), (6, 40), (7, 60) >$. It is important given that the NB value should carefully determined If NI is large in this example, we may have many duplicate split time series.

3.3   Time Series Normalization

After splitting a time series, the obtained data $A' = \{a'_k\}$ and the query time series $qdata$ are normalized in terms of time and price.

When normalizing the price of a time series, we produce a price- normalized time series based on the maximum price of all the final prices in the auction time series in the same category as $qdata$. For an element of $(T'_k(j), p'_k(j))$ of $\{a'_k\} \in A'$, we calculate $p'_k(j)/maxprice$, where $maxprice = max_i\{p_i(NB_i)\}$. Then, we construct the time normalized time series d'_k from a'_k, where d'_k is is a sequence of $(T'_k(j), p'_k(j)/maxprice)$.

A time series is also normalized based on its closing time. We introduce an interpolation function $f(t_s, d'_k)$. The function is a function that estimates an interpolation value at a intermediate time point of a time series d'_k. A time point t_s is in $[0.0, 1.0]$ Using this function, we transform a sequence d'_k to d_k with time- normalized time series as follows. We have the number of time points in $[0.0, 1.0]$ and a set of time points $T = \{t_s = (s - 1) * \delta | s = 1, .., NT\}$, where $\delta = 1.0/(NT - 1)$. We then construct $d_k = < d_k(1), d_k(2), .., d_k(s), .., d_k(NT) >$ where $d_k(s) = f(t_s, d'_k)$. d_k is obtained as follows: for each $t_s \in T$, $f(t_s, d'k)$ is calculated and set to $d_k(s)$. Any interpolation function such as B-spline and Bezier, can be used as the function f.

For example, $a = < (0, 4), (3, 30), (5, 34), (6, 40), (7, 60) >$ can be normalized such that $maxprice = 100$. The price- normalized time series d' is $< (0, 0.04), (3, 0.30), (5, 0.34), (6, 0.40), (7, 0.60) >$. When $NT = 6$, a normalized time series $< (0.0, f(0.0, d')), (0.2, f(0.2, d')), (0.4, f(0.4, d')), (0.6, f(0.6, d')), (0.8, f(0.8, d'))(1.0, f(1.0, d')) >$ is obtained. As the time series has the uniform interval and the interval is common to all the normalized time series, the normalized time series data obtained d can be represented $< f(0.0, d'), f(0.2, d'), f(0.4, d'), f(0.6, d'), f(0.8, d'), f(1.0, d') >$.

3.4   Clustering

The clustering produces a set of clusters $C = \{c_m | m = 1, .., NC\}$ from a set of normalized time series including $qdata$. Any clustering method can be applied for the clustering of the time series and any distance calculation method between two time series data can be used, but the typical non-hierarchical clustering method known as k-means and the conventional Euclidean are used to obtain

the clusters in this method. The number of the clusters, NC, needs to be given when using k-means. Based on our preliminary evaluations, we set NC to a value from 5 to 10. After clustering, the cluster $c_q \in C$ containing $qdata$ is identified and checked. The final price estimation is achieved by calculating $average(P_i)$, where P_i is the final price of the auction related to $d_i \in c_q$ ($d_i \neq qdata$). If there is no other auction time series other than $qdata$ in c_q, the average of the previous final prices cannot be calculated. In this case, the average of the final prices of all the auction time series A related to the $qdata$, is calculated.

4   Evaluations

4.1   Test Data

We used two kinds of test dataset for our evaluations: Yahoo! auction and eBay dataset. The first of the test dataset is extracted from Yahoo! auction[1], the most popular auction web service in Japan. The auction items used as test data are selected to meet the following conditions:

– The number of previous auctions of a product was more than 60.
– The standard deviation of the final price of a product ranged from 2.0 to 20.0.
– No considerations on auction parameters and product characteristics were taken into consideration on test data selections,

Based on these conditions, five products were selected as test data. We selected a subset of test data from the past auctions at random while the rest were selected as a query auction data set to predict their final prices. We cut the query time series from the auction starting time up to the current time $T_{current}$ before

Table 2. Test data

Product	Product name	Final price range (Yen)	Average final price (Yen)	Std. Dev. of final prices	No. of past items	No. of items to predict
G1	iPad2 Wifi 3G,64GB	66,000 - 99,800	87,634	10,173	75	13
G2	Herman Miller Aeron Chairs AE113AWB	40,500 - 105,000	56,003	16,652	185	40
G3	Toshiba REGZA-32RX1	34,800 - 70,001	87,634	7,143	94	26
G4	Onepiece Complete	15,000 - 25,800	19,005	3,196	62	10
G5	Ricoh Camera CX1	5,000 - 13,500	8,725	2,423	80	11

[1] http://auctions.yahoo.co.jp/

the auction closing time . $T_{current}$ was set 70% of the auction closing time. The details of the test data are shown in Table 2.

The other test dataset is the complete bidding records for 149 auctions on Palm M515 PDAs ion eBay[2]. All the auctions in this dataset is 7-day auctions and transacted between March and June in 2003. A set of only time series extracted from randomly selected 80% of the eBay dataset was used as past auction histories. Prediction experiments were conducted for the remaining 20%.

4.2   Evaluation Results Using Yahoo! Dataset

We performed three experiments for Yahoo! data set as follows.

– Accuracy experiment. Our proposed method was compared with an actual prediction web service and existing prediction methods using the test data described above. The web service used for the comparison is known as Rakuboz [3], one of the web services of the auction final price estimations made for Yahoo! auction. The two existing method were the curve fitting method using $Price = \alpha_2 Time^{\alpha_1}$ proposed by Kurosawa et al. [12] and the naïve method which only calculates the average price. The measure used for the comparison was defined by the following expression: $Accuracy = 1 - |Price_{predicted} - Price_{actual}|/Price_{actual}$. Note that $1 - Accuracy$ is the mean absolute percentage error (MAPE). In this experiments, we used the proposed method without splitting to evaluate the raw performance of the method. The number of clusters (NC) is 5 in this experiment.
– Splitting effect experiment. We made some experiments to check S effects on the accuracy of our method. We set S={70,100} by adding a set of data split at a point of 70% with the previous test data.
– Prediction time effect experiment. We changed the time $T_{current}$ when the prediction was performed. We assumed that the conventional prediction time was a time when 70% of the overall auction period had elapsed from the starting time. In this experiment, we varied the time-point from 50% until 90% in increments of 10%. This evaluation determined whether the current auction was classified into the correct clusters by checking the clustering results by hands. For all the query auction time series for the same product, we calculated the number of queries that were classified into the correct cluster. We used the same method as in the first experiment, i.e. the method without splitting.

Accuracy Comparison. Table 3 shows the result of the accuracy comparison in the first experiment and also shows the standard deviations for the accuracy results.

---

[2] Available at http://www.rhsmith.umd.edu/digits/statistics)
[3] http://www.rakuboz.com/

Table 3. Results of Accuracy (left) and Standard deviations(right)

Prd.	Our method	Rakuboz	Curve fitting	Naïve	Prd.	Our method	Rakuboz	Curve fitting	Naïve
G1	89.02	63.40	85.67	88.92	G1	7.53	32.89	7.97	9.12
G2	82.97	58.01	83.22	79.07	G2	9.25	35.97	13.41	18.83
G3	88.34	63.98	87.99	87.40	G3	7.43	33.28	5.06	10.63
G4	88.40	52.23	84.56	85.46	G4	5.71	36.24	5.38	14.65
G5	84.35	61.20	86.21	75.19	G5	6.68	37.49	12.87	7.79
Average accuracy	86.62	59.76	83.21	85.53	Aveerage std. dev.	7.32	35.17	8.94	12.20

We made the following observations.

- Our proposed method was superior to the other three methods in terms of accuracy. The average accuracy was the highest with our method, 86.63%. The 2nd highest accuracy, 85.53% was obtained by Naïve method, the total average price calculation.
- Our method produced the lowest standard deviation for the accuracy, 7.32%. The 2nd smallest standard deviation of accuracy, 8.94% is obtained by Curve fitting method [12]. Our method is considered to be stable from a viewpoint of this smallness of the standard deviation.
- Rakuboz, the actual price prediction web site, produced the worst accuracy and the highest standard deviation. This was because there were several cases when Rakuboz did not produce final price predictions. We set 0.0 as the predicted price in these cases.

Splitting Effect. In the case when a split time series is mixed with the original time series, the result is shown in Table 4. As shown in this table, the average accuracy for {70,100} was 87.11%, which was 0.49% higher than the method without splitting. The average standard deviation for {70,100} was 6.67%, which was 0.65% lower than 7.32%, achieved with the method without splitting. Therefore, we conclude that the method with splitting can produce more accurate and stable predictions, compared with the method without splitting.

However, it should be noted that the number of clusters (NC) for {70,100} was necessary to be larger than NC for {100}, because the splitting produces more data to be clustered, which was twice in this case. We should remark that NC needs to be chosen carefully. It is because a wrong NC setting may produce a worse result, observed from the result of the case S={100} in Table 4. The optimal NC determination is an issue to be remained. The optimal splitting is another issue to be solved. Further splittings, such as {60, 70,100} and {70, 80,100}, will also be necessary to be evaluated in future.

$T_{current}$ **Effect.** Fig.5 shows the result of the experiment, where we observed the effect of $T_{current}$ on predictions. The horizontal axis represents the percentage of prediction time. from the start until the closing time. The percentage was

Table 4. S and NC effects (Accuracy (left) and Standard deviations(right))

	S={70,100} (NC=7)	S={100} (NC=7)		S={70,100} (NC=7)	S={100} (NC=7)
G1	91.20	89.30	G1	4.45	7.79
G2	81.94	79.53	G2	9.94	13.44
G3	89.20	86.66	G3	5.89	8.65
G4	87.96	87.9	G4	6.45	7.71
G5	85.24	79.84	G5	6.64	10.82
Ave. accuracy	87.11	84.65	Ave. std. dev.	6.67	9.68
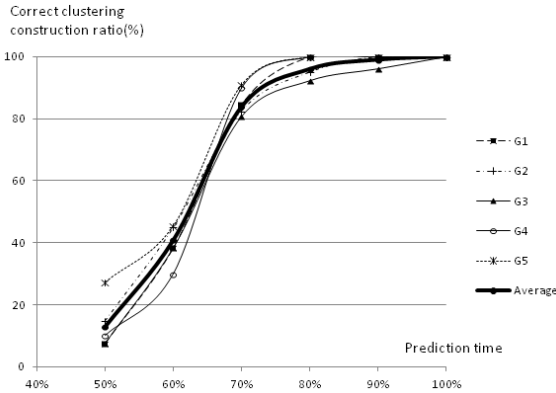
Fig. 5. Prediction time effect

varied from 40% to 90% in the experiment. The vertical axis represents the percentage where the current auction time series data was clustered into the correct cluster. The bold line in this figure is the average of five product results while the other thin lines are the individual product results. At 70%, the percentage of correct cluster constructions was determined about 84%, showing that the clustering was performed appropriately. Based on this figure, it was shown that the 70% time-point was reasonable for accurately estimating the auction final price.

4.3   Results Using eBay Dataset

We conducted an accuracy experiment using eBay dataset. The result is shown in Table 5. As shown here, our method with S={70,100} was better than the curve fitting method and Naïve method. Moreover, it was obtained that our method with splitting was better than the method without splitting, on the eBay dataset.

Table 5. Results using eBay dataset

	Our method S={70,100} NC=5	Our method S={100} NC=5	Curve fitting	Naïve
Average accuracy	91.45	89.22	85.39	87.61
Average standard deviation	5.12	7.56	11.3	7.81

5   Related Work

There are many previous studies of online auctions, which are mainly segregated into auction analysis and auction final price prediction. Research on the latter is briefly summarized below due to space limitations. Methods used for final price prediction include Grey system theory [5], Functional data analysis [4], Parametric growth curves [6], Regression trees [1], Autoregressive (AR) [7], Curve clustering [11], Beta model [8], Bayesian networks [2], K-nearest neighbor [9] [10] and Neural networks [3]. No previous methods have combined the splitting of auction histories with clustering. Please refer to an excellent survey on online auction final price prediction from a statistical modeling viewpoint for further information [13].

6   Conclusion

In this paper, we propose a novel auction final price prediction method using splitting and clustering to realize more accurate estimation of final prices. The important points of the method are 1) the use of clustering and 2) the splitting of auction histories. Clustering allows us to segregate a set of past auction histories for the same product. All the auction time series in a cluster including the current auction which are obtained by clustering of the time series, are much similar to the current auction time series. Therefore, the prediction is more accurate because it is based on aggregation of the similar time series rather than all the related time series. Splitting a time series causes more time series to be clustered. Thus, more time series patterns can be considered in the final price prediction. Our experiments showed that the proposed method was superior to an actual price estimation web service and some existing prediction methods.

There are several ways of extending our proposed method. First, we need to obtain more accurate prediction in the early stage of auctions. In this paper, we predicted the final price with 86.62% accuracy, at a time-point when the 70% after the starting time. We need to improve the accuracy in predicting at an early time-point. Moreover, we need to evaluate our method in more detail and to compare our method with other final price prediction methods. Finally, we intend to develop a new auction final price prediction web service.

References

1. Heijst, D.V., Potharst, R., Wezel, M.V.: A support system for predicting eBay end prices. Decision Support Systems 44, 970–982 (2008)
2. Hongil, K., Seung, B.: Forecasting Winning Bid Prices in an Online Auction Market -Data Mining Approaches. J. of El. Sci. & Tech. of China 2(3), 6–11 (2004)
3. Ghani, R., Simmons, H.: Predicting the End-Price of Online Auctions. In: ACM ECML/PKDDD (2004)
4. Wang, S., Jank, W., Shmueli, G.: Explaining and Forecasting Online Auction Prices and Their Dynamics Using Functional Data Analysis. J. of Business and Economic Statist 26(2), 144–160 (2008)
5. Lim, D., Anthony, P., Ho, C.M.: Agent for Predicting Online Auction Closing Price in a Simulated Auction Environment. In: Ho, T.-B., Zhou, Z.-H. (eds.) PRICAI 2008. LNCS (LNAI), vol. 5351, pp. 223–234. Springer, Heidelberg (2008)
6. Hyde, V., Jank, W., Shmueli, G.: A family of growth models for representing the price process in online auctions. In: ACM ICEC 2007, pp. 399–408 (2007)
7. Jank, W., Shmueli, G., Wang, S.: Dynamic, Real-time Forecasting of Online Auctions via Functional Models. In: KDD 2006, pp. 580–585 (2006)
8. Jank, W., Shmueli, G., Zhang, S.: A flexible model for estimating price dynamics in on-line auctions. J. of Royal Stat. Soc., Appl. Stat. 59, Part 5, 781–801 (2009)
9. Zhang, S., Jank, W., Shmueli, G.: Real-time forecasting of online auctions via functional K-nearest neighbors. Int. J. of Forecasting 26, 666–683 (2010)
10. Raykhel, I., Ventura, D.: Real-time Automatic Price Prediction for eBay Online Trading. In: AAAI 2009, pp. 135–140 (2009)
11. Jank, W., Shmueli, G.: Profiling Price Dynamics in Online Auctions Using Curve Clustering, Robert H. Smith School Research Paper No. RHS-06-004 (2005)
12. Kurosawa, S., Maekawa, T.: The analysis of bidder's behavior of the Internet auction. IPSJ, SIG-EIP 2001(118), 7–14 (2001) (in Japanese)
13. Shmueli, G., Koppius, O.: The Challenge of Prediction in Information Systems Research. Social Science Research Network Working Paper Series (March 2008)
14. NIST/SEMATECH e-Handbook of Statistical Methods,
    http://www.itl.nist.gov/div898/handbook/

Taxonomy-Oriented Recommendation towards Recommendation with Stage⋆

Lei Li[1], Wenxing Hong[2], and Tao Li[1]

[1] School of Computer Science, Florida International Univ., Miami, FL 33199, USA
{lli003,taoli}@cs.fiu.edu
[2] Automation Department, Xiamen University, Xiamen, P.R. China 361005
hwx@xmu.edu.cn

Abstract. In some E-commerce recommender systems, a special class of recommendation involves recommending items to users in a life cycle. For example, customers who have babies will shop on *Amazon* within a relatively long period, and purchase different products for babies within different growth stages. Traditional recommendation algorithms cannot effectively resolve the situation with a life cycle. In this paper, we model users' behavior with life cycles by employing hand-crafted item taxonomies, of which the background knowledge can be tailored for the computation of personalized recommendation. In particular, our method first formalizes a user's *long-term behavior* using the item taxonomy, and then identify the *exact stage* of this user. By incorporating collaborative filtering into our method, we can easily provide a personalized item list to this user through other similar users within the same stage. An empirical evaluation conducted on a purchasing data collection obtained from *Amazon* demonstrates the efficacy of our proposed method.

Keywords: Taxonomy, Recommendation with Stage, Long-Term, Short-Term.

1 Introduction

Personalized recommendation is becoming increasingly popular due to the great utility in providing people with products that they might be interested in and thus purchase. Many electronic commerce websites in a business-to-consumer (B2C) style, e.g., *Amazon*, have already benefited from personalized marketing leverage offered by product recommender systems [16,18]. Most researchers try to develop recommender systems by utilizing content-based methods or collaborative filtering or hybrid version of these two techniques. However, such recommendation paradigms cannot effectively handle all the cases that might happen in real-world applications.

In E-commerce recommender systems, there is a special class of recommendation problems, in which a user's behavior might evolve over time, i.e., within different stages the user will prefer distinct items, as shown in *Scenario 1*.

Scenario 1: A customer with a baby needs to purchase some products of baby care. Within different growth stages of his/her baby, the desired products are significantly different.

In this scenario, the customer's purchasing interest evolves over time. It is not reasonable to utilize item-based method for personalized recommendation, as the items purchased in different time periods might significantly different. Further, it is a non-trivial task to effectively capture the evolution of customer's purchasing interest. In general, an item taxonomy is often associated with a recommender system, by which customers can easily navigate different categories of products. In this paper, we propose to employ such taxonomic knowledge to formalize users' long-term preference, and in the meantime, to capture the evolution of users' interest. However, only long-term preference cannot provide enough evidence against the user's current desire, as illustrated in *Scenario 2*.

Scenario 2: A customer with a 6-month baby needs to change the type of formula for his/her baby.

In this scenario, the customer needs to purchase some new items. In general, long-term preferences model users' purchasing interest in a long run, e.g., what brands that the user prefers. However, for customers who step into a new purchasing stage, he/she would desire new types of items that have never been purchased before by this user. Therefore, only considering the customer's long-term preference will not capture his/her desire. To handle such issue, we propose to explore the user's short-term interest by analyzing similar users' behaviors. Here "similar" indicates that two users might have similar purchasing behaviors within the identical purchasing stage. By this way, we can easily resolve the situation that the customer desires new types of products within new stages.

In summary, the contribution of our paper is three-fold:

- We define a new class of recommendation problem, named *Recommendation with Stage* (RwS), in which a user's preference evolves over time.
- We introduce a taxonomy-oriented approach to model a user's preference. We propose to model a user's preference not only on item level, but also on the semantic correlations among the categories that the items belong to.
- We propose a novel graph-based model for recommendation, in which a multi-modal weighted graph, including users, items and concepts, is constructed, and then random walk is performed on this graph for inference.

The paper is organized as follows. We describe some related work in Section 2. Section 3 defines the formal statement of the problem. In Section 4, we discuss the algorithmic details of our proposed method. Section 5 provides an empirical evaluation on a real-world data set, and finally Section 6 concludes the paper.

2   Related Work

In this section, we briefly review prior methods related to recommender systems. We also highlight the difference between our method and other existing work.

Content-Based Recommendation. The effectiveness of content-based information filtering paradigm has been proven for applications locating textual

documents relevant to a topic. Particularly in recommender systems, content-based methods [1,4,7] enable accurate comparison between different textual or structural items, and hence recommend items similar to a user's consumption history. However, content-based filtering approaches suffer from multiple drawbacks, e.g., strong dependence on the availability of content, ignoring the contextual information of recommendation, and etc.

Collaborative Filtering. Collaborative-based recommenders tackle the recommendation problem by exploiting similar users' profiles and recommending new items not previously rated or seen by the target user. Recommendation is therefore achieved by finding common characteristics in the preferences of other users. Common approaches in this category include k-Nearest-Neighbor (k-NN), matrix factorization [8,15], and etc. *Amazon* uses an item-to-item collaborative filtering approach to personalize the online store for each customer [11]. However, collaborative filtering requires more ratings over items, and therefore suffers from the rating sparsity problem.

Hybrid Recommendation. Hybrid approaches combine the aforementioned techniques in different ways to improve recommendation performance and tackle the shortcoming of underlying methods. Some sample approaches include [3,9], in which the inability of collaborative filtering to recommend items is commonly alleviated by combining it with content-based filtering.

Knowledge-Based Recommendation. Knowledge-based personalization systems emphasize explicit domain knowledge about the items as well as implicit knowledge about the users (e.g., psychographic, demographic, or other personal attributes of users) to extract pertinent recommendations [6,17]. Some researchers use item taxonomies to raise the accuracy of the results [2,5,12].

In our work we investigate a special class of recommendation problems, in which a user's preference evolves over time. We formalize the user's preference by virtue of the category correlations within item taxonomies. To provide attractive recommendation list, we first identify the exact purchasing stage that the user lies in, and then localize the user's short-term preference by inference over users, items, and concepts, where the candidate users' short-term profiles that are within the same stage compared with the target user are selected to construct the inference base.

3   Problem Formation

3.1   Background

In our work, we employ item taxonomy to facilitate the recommendation procedure. In general, item taxonomies are often designed or applied in E-commerce websites, e.g., *Amazon*. In particular, we have the following elements within an E-commerce community:

- **User Repository:** $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$. $\mathcal{U}$ contains all users or customers within the E-commerce community.

- **Product Repository:** $\mathcal{P} = \{p_1, p_2, \cdots, p_m\}$. $\mathcal{P}$ is composed of all the products provided by the E-commerce website.
- **User Rating:** $\mathcal{R} = \{R_1, R_2, \cdots, R_n\}$. $\mathcal{R}$ includes all the rating information of users over products.
- **Item Taxonomy:** $\mathcal{T}$ over item category set $\mathcal{C} = \{c_1, c_2, \cdots, c_l\}$. The taxonomy $\mathcal{T}$ captures the hierarchical structure of $\mathcal{C}$ in the E-commerce. For simplicity, we require that one product can only fall into one specific category, i.e., $\{p \to c_i | \neg c_j, c_j \neq c_i, p \to c_j\}$.

3.2   Problem Statement

Problem (RECOMMENDATION WITH STAGE): *Within an E-commerce community, given $\mathcal{U}$, $\mathcal{P}$, $\mathcal{R}$ and $\mathcal{T}$, recommend items to a target user, guaranteeing that the user's current purchasing desire is maximally satisfied.*

To resolve the above problem, we can decompose it into two different subproblems based on the characteristics of the problem itself.

Subproblem 1 (STAGE IDENTIFICATION): *Given an item taxonomy and a target user's history, identify the user's current preference on item categories.*

To identify the recommendation stage for the target user, we need to consider the entire purchasing history of this user. Simply modeling the user's behavior based on transactional purchasing data cannot capture the evolution of the user's exact interest. Fortunately, the item taxonomy can provide us a meaningful and elegant explanation of the user's evolved behavior.

Subproblem 2 (ITEM RECOMMENDATION): *Given a target user's current stage, recommend a list of items to the user such that his/her preference will be maximally satisfied.*

Once we obtain the recommendation stage of the target user, a natural way to recommend items is to refer other users' behaviors within the same stage. In our problem setup, we have additional resources to use, e.g., the item taxonomy, to enrich the correlations among users, and therefore we are able to provide more reasonable recommendation list.

4   Taxonomy-Driven Recommendation

The item taxonomy semantically encapsules the correlations among items, which is suitable to capture users' preferences over different item categories. Specifically in our work, we first separate a user's purchasing history into different stages, and then model the user's long-term preference by employing the item taxonomy. Given a target user, we construct a multi-modal graph based on the users' purchasing behaviors similar to the target user, and then perform inference on this graph to finalize the recommendation.

4.1   Stage Identification

The long-term user profile consists of multiple stage profiles. Formally, given the purchasing history $\mathcal{H}$ of the target user u, $\mathcal{H}$ can be initially divided into

multiple segments based on a specific time scheme. Here the time scheme used to segment $\mathcal{H}$ can be learned according to the detailed content of $\mathcal{H}$ using statistical or machine learning techniques. Taking into account the time complexity, hereby, we simply define a uniform time frame T as a segment unit, and then separate $\mathcal{H}$ based on T, i.e., $\mathcal{H} = \{H_{t_0}, H_{t_1}, \cdots, H_{t_n}\}$, where t_0 means the current stage, i.e., the latest purchasing period. For each time period t_i, $i = 0, 1, \cdots, n$, we model the user's behavior H_{t_i} using the item taxonomy.

User Profile Generation. As illustrated in Fig 1, to model the stage profile $\mathcal{H}_{t_0}$ of user u_1, we first extract the transactional data of u_1, $R_1 = \{p_{41}, p_{51}, p_{21}\}$. For each transaction or item in R_1, e.g., p_{41}, we locate the fine-grained category that the item directly belongs to, e.g., c_4, and then traverse the taxonomy from bottom to top, to obtain the category path starting from this category, e.g., $c_4 \rightarrow c_1$. Following this example, we then transfer the rating score on p_{41} to the category path, i.e., assigning "1" to each category in the path. Subsequently, we aggregate the rating score for each category and formalize the user's stage profile as a weighted category vector.
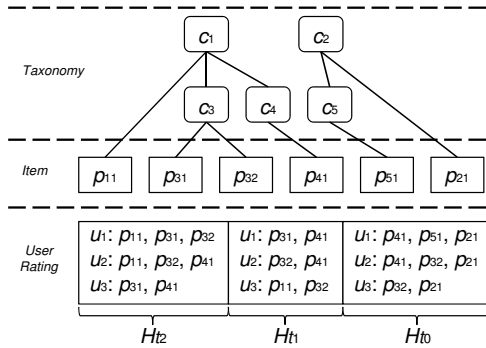

Fig. 1. An example of item taxonomy

Again for the previous example, assume the basic category vector is in the form of $\langle c_1, c_2, c_3, c_4, c_5 \rangle$, then the weighted category vector $R_{u_1}^{t_0} = \langle 1, 2, 0, 1, 1 \rangle$, where each entry represents the user's implicit rating over the corresponding category. In such representation, we assign more weight on the categories of the upper level in the item taxonomy, by which we can avoid being trapped into specific topics and losing the overall recognition of the user's preference. The weighted category vector is l_2-normalized. Finally, we can denote user u_1's long-term profile as $\mathbb{R}_{u_1} = \{R_{u_1}^{t_0}, R_{u_1}^{t_1}, R_{u_1}^{t_2}, \cdots\}$, and also we can identify the current recommendation stage of u_1 as H_{t_0}.

The identification of the user's current recommendation stage can help determine what information should be used in the item recommendation, i.e., to consider other users' stage profiles. For example, there are three users u_1, u_2 and u_3, whose long-term profiles are $\{R_{u_1}^{t_0}, R_{u_1}^{t_1}, R_{u_1}^{t_2}\}$, $\{R_{u_2}^{t_0}, R_{u_2}^{t_1}, R_{u_2}^{t_2}, R_{u_2}^{t_3}, R_{u_2}^{t_4}\}$, and

$\{R_{u_3}^{t_0}, R_{u_3}^{t_1}, R_{u_3}^{t_2}, R_{u_3}^{t_3}\}$. Given the target user u_1 with the recommendation stage H_{t_0}, we need to analyze the stage profiles of users u_2 and u_3, that is, $R_{u_2}^{t_2}$ and $R_{u_3}^{t_1}$, since these stage profiles indicate that their stages are identical to u_1's current stage. In this sense, the long-term and short-term profiles of a user are dependent in a way that the short-term profile is derived from the long-term profile and also the long-term profiles of different users can contribute to the success of collaborative filtering. The subsequent procedures are all based on this scheme. Note that the evolution of a user's purchasing interest exists within the user's long-term profile, whereas the interest would be relatively stable within the user's short-term profile.

Model Refinement. As proposed in the previous section, the item taxonomy serves to connect users and items in a semantic way, in which users' rating behaviors are amplified via the category path. In such a model, we can easily capture the correlations between different elements, i.e., users, items and categories. In the following, we further formalize the taxonomy-based profiling model by specifying three different similarities.

- *User-User Similarity* ($\mathcal{S}_U$): The user-user similarity originates from two different components: user-item similarity S_{UI} and user-category similarity S_{UC}. Given two users u_1 and u_2, S_{UI} can be computed by considering the *Jaccard similarity* between the item sets purchased by these two users, I_{u_1}, I_{u_2}, whereas S_{UC} can be derived from the *Cosine similarity* between the identical stage profile vectors of these two users R_{u_1}, R_{u_2}. Specifically,

$$S_{UI}(u_1, u_2) = \frac{I_{u_1} \cap I_{u_2}}{I_{u_1} \cup I_{u_2}}, S_{UC}(u_1, u_2) = \frac{R_{u_1} \cdot R_{u_2}}{||R_{u_1}||||R_{u_2}||}.$$

Finally, the similarity between two users, $S_U(u_1, u_2)$ can be calculated as:

$$S_U(u_1, u_2) = \lambda \cdot S_{UI}(u_1, u_2) + (1 - \lambda) \cdot S_{UC}(u_1, u_2). \tag{1}$$

In this way, we can easily capture the correlations between two users from both the taxonomic level and real purchasing behavior level simultaneously.

- *Item-Item Similarity* ($\mathcal{S}_I$): Similarity between items can be computed using item-to-item collaborative filtering. Given a product p, our goal is to find products similar to p; Here by "similar" we mean that the user sets, U_{p_1}, U_{p_2}, purchasing the two items have substantial overlap. To this end, we use the *Jaccard similarity* to capture $S_I(U_{p_1}, U_{p_2})$, as

$$S_I(U_{p_1}, U_{p_2}) = \frac{U_{p_1} \cap U_{p_2}}{U_{p_1} \cup U_{p_2}}. \tag{2}$$

- *Category-Category Similarity* ($\mathcal{S}_C$): This similarity can help quantify the semantic correlation between two different categories in the item taxonomy. In our work, we employ *Information Content* (IC) [14] to compute $\mathcal{S}_C$, which measures the amount of information of a given category c from its probability of occurrence, in our case, the probability that items under c are

purchased. The larger the IC, the more popular the category. IC can be computed as the negation of the logarithm of the probability of occurrence, i.e., $IC(c) = -\log p(c)$. To derive the similarity between two categories, we use the similarity measure proposed in [10], which evaluates the correlation between a pair of concepts as the IC of the *Least Common Subsumer* (LCS) in a give taxonomy, i.e., an indication of the amount of information that two categories share in common. Given two categories c_1 and c_2, the similarity $S_C(c_1, c_2)$ can be computed as

$$S_C(c_1, c_2) = \frac{2 \times IC(LCS(c_1, c_2))}{IC(c_1) + IC(c_2)}. \tag{3}$$

4.2   Item Recommendation

Once we locate the recommendation stage of the customer, a simple solution to the recommendation is to employ user-to-user collaborative filtering to retrieve items that the customer might prefer. However, such paradigm might result in the overemphasis on popular items, and thus make popular items more popular, while the items in the long tail have little chance to be recommended to the user. To resolve this issue, we investigate the effect of using the item taxonomy to increase the possibility that items in the long tail are being recommended. To this end, we first construct a multi-modal graph, and then given a target customer u, we perform random walk on this graph starting from u for recommendation.

In detail, the multi-modal graph consists of multiple resources, e.g., users, products and categories, associated with multiple types of correlations, e.g., user-user, item-item, category-category relations and etc. These resources are all originated from the user's purchasing behaviors within the same stage. We encapsule the graph as an adjacency matrix. Formally, let U, P, C denote customers, products and categories respectively, and let U_p, U_c, P_c denote user-product, user-category and product-category relations, then we can build a block-wise adjacency matrix $\mathbb{W} = \begin{pmatrix} U_u & U_p & U_c \\ U_p & P_p & P_c \\ U_c & P_c & C_c \end{pmatrix}$, where U_u, P_p, C_c denote user-user, product-product and category-category relations. Here the entries of U_u is calculated by S_U defined by Eq.(1), P_p by Eq.(2) and C_c by Eq.(3). The entries of other blocks in $\mathbb{W}$ can be derived by purchasing behaviors or the item taxonomy. For example, for entries in U_p, each one is a binary value ("1" or "0"), representing whether the corresponding user has been purchasing the product or not.

Given a target user u, we perform random walk on the multi-modal graph starting from u. Here we employ *random walk with restart* (RWR) [13] to retrieve items for recommendation, such that the recommended items are not deviated much from the user's purchasing interest. Specifically, we construct a matrix $\mathbb{A}$ using normalized graph Laplacian, i.e., $\mathbb{A} = D^{-1/2} \mathbb{W} D^{-1/2}$, in which D is a diagonal matrix with its (i, i)-element equal to the sum of the i-th row of $\mathbb{W}$. The algorithmic detail is described in Algorithm 1.

In Algorithm 1, we first build a multi-modal graph whose nodes include users, products and categories, and the weighted edges between nodes are quantified

---

Algorithm 1. RECOMMENDATION USING RWR

---

input: An adjacency matrix $\mathbb{W}$, and a target user u_q.
output: A recommended item list l.

1. Let $\overrightarrow{\mathbf{v_q}} = 0$ for all its entries, except a '1' for the q-th entry;
2. Normalize $\mathbb{W}$ using $\mathbb{A} = D^{-1/2}\mathbb{W}D^{-1/2}$;
3. Initialize $\overrightarrow{\mathbf{u_q}} = \overrightarrow{\mathbf{v_q}}$;
4. **while** $\overrightarrow{\mathbf{u_q}}$ not converged **do**
5. $\overrightarrow{\mathbf{u_q}} = (1 - \delta)\mathbb{A}\overrightarrow{\mathbf{u_q}} + \delta\overrightarrow{\mathbf{v_q}}$;
6. **end while**
7. Select top ranked subset $\mathbf{e}$ from $\overrightarrow{\mathbf{u_q}}$ as the recommendation base;
8. **if** e_i is a user **then**
9.   Select items that e_i has purchased recently and put them into l;
10. **end if**
11. **if** e_i is a product **then**
12.   Put e_i into l;
13. **end if**
14. **if** e_i is a category **then**
15.   Select items that contribute more to IC of e_i and put them into l;
16. **end if**
17. **return** l;

---

by the adjacency matrix $\mathbb{W}$. Here we normalize $\mathbb{W}$ using graph Laplacian (line 2) to make it suitable for the random walk computation. Next, our method executes RWR (line 3-6) to retrieve elements highly related to a given user q. The procedure will end once the input vector $\overrightarrow{\mathbf{u_q}}$ converges. Here the parameter δ controls the steps of each random walk; in our case, we expect that the random walk be not so deep on the graph so that the selected items will conform to the user's purchasing interest.

After $\overrightarrow{\mathbf{u_q}}$ is finalized, our method choose a subset of elements that are top ranked in $\overrightarrow{\mathbf{u_q}}$, which might include users, products or categories. Therefore, to retrieve the recommended items, we need to consider three distinct cases (Line 8-16). In this way, the resulted recommendation list is composed of products that are originated from different recommendation disciplines.

5   Empirical Evaluation

5.1   Real-World Dataset

The dataset used in our experiment is collected from *Amazon.com*. It consists of customers' order information related to *Baby Care*, ranging from Jan 21st, 2005 to Mar 5th, 2009. This dataset contains 133,039 orders of 1,000 anonymous customers on 2,187 products. The ratings in this dataset are implicit rating, i.e., the binary rating originated from the purchasing behavior of customers. The time span of the customer's order history varies significantly, ranging from several months to 4 years. The average number of orders for customers is 133.

The item taxonomy covered by the dataset consists of 37 classes and it is a three-layer hierarchy: the root class, "Baby Care" and subclasses, e.g., "Baby Diaper", "Baby Formula", "Feeding Accessories", and etc. For each category, the number of products contained varies from tens to hundreds. Hereby, the taxonomy is simple and items ratings in the dataset are relatively dense. Due to the space limit, we are unable to list all the classes contained in the taxonomy.

5.2   Experiments

Profile Generation and Taxonomy. Long-term and short-term profiles both contribute to the success of recommendation. To verify this, we use the long-term and short-term profiles, respectively, to perform the recommendation task. Also in our framework, we employ the item taxonomy to enrich the representation of users' profiles. Therefore, we setup four baseline methods as follows: (1) Long-term without taxonomy (LT): use each customer's entire purchasing history to construct profiles and perform collaborative filtering; (2) Short-term without taxonomy (ST): use each customer's recent purchasing history[1] to build profiles and perform collaborative filtering; (3) Long-term with taxonomy (LTT): use item taxonomy to enrich each customer's long-term profile and perform collaborative filtering; (4) Short-term with taxonomy (STT): use item taxonomy to enrich each customer's short-term profile and perform collaborative filtering. In addition, we implement two existing approaches, [18] (TDC) and [12] (EUI) for comparison, where the former considers both user proximity and item proximity, and the latter uses taxonomy to recommend novel items to users.

   For evaluation of our method (RwS for short), we empirically set the time frame T introduced in Section 4.1 to be 90 days, and use the purchasing history within the latest stage as the testing data, whereas the purchasing history before the latest stage is regarded as the source of profiling. Note that in our method, user similarities are calculated based on the stage closest to the target user's current stage. The parameters λ and δ will be tuned in the following subsection. For each approach, we recommend products (top@10, top@20 and top@30) to all the users in the dataset, and then compute the averaged precision, recall and F1-score of the recommended list. Fig 2 shows the comparison results.

   From the results, we observe that (i) The performance of algorithms with item taxonomy involved is superior to the ones without taxonomy. It is evident that by using item taxonomy, users' preference can be better captured, since the categories or classes have more distinguishable power than simple items on identifying a user's purchasing interest. (ii) By considering the short-term profiles, the accuracy of the recommended item list can be slightly improved, which is straightforward because we are more concerned with users' recent activities. (iii) Our proposed method significantly outperforms other candidates, which demonstrates the efficacy of our algorithm in handling the problem of RwS.

---

[1] Here we empirically set the "recent" history as the last 90 days behavior, and when performing collaborative filtering, we only consider all customers' recent histories.
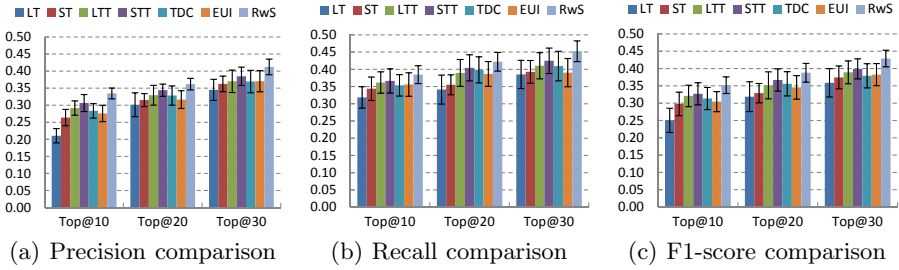
(a) Precision comparison   (b) Recall comparison   (c) F1-score comparison

Fig. 2. Performance comparison for different recommendation algorithms

Model Validation. In Section 4.1 we introduced three distinct similarity mea-
surements, i.e., *user-user*, *item-item* and *concept-concept*. The encapsulation of
these three measures enables the method to capture the correlations among dif-
ferent resources (users, items and categories), and therefore renders the recom-
mendation more reasonable. Also, the item taxonomy can help better interpret
the customers' exact purchasing interest. To verify this claim, we evaluate the
performance of using different measurements. In the experiment, we choose to
use user-user and item-item similarities respectively, as two baselines, and then
recommend items (top@10, top@20 and top@30) to a set of randomly selected
users (100 users). Note that the parameters λ and δ are set to be 0.2 and 0.7,
respectively. Detailed parameter tuning process can be found in next section.
We plot the average precision and recall pair of recommendation results. As
depicted in Fig 3, besides the higher ratio of precision and recall, the perfor-
mance distribution of the unified multi-modal graph is more compact than the
others, which demonstrates the effectiveness and stability of the strategy for
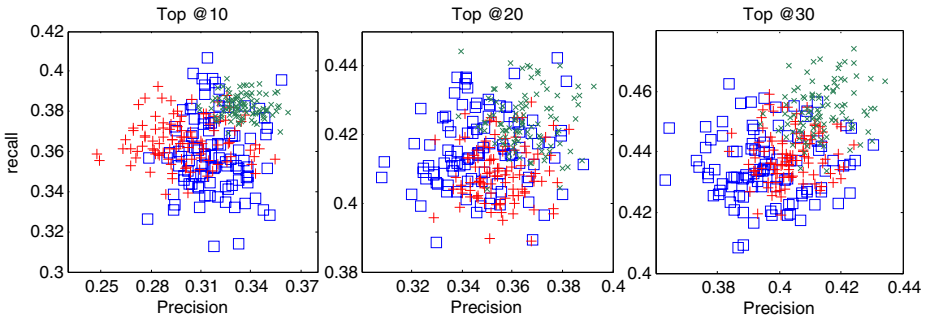multi-modal graph construction.

Fig. 3. Precision-recall plot for different model constructions. Remark: "□" represents
the performance using user-user similarity; "+" denotes the performance using item-
item similarity; and "x" represents the one using the integration of user-user, item-item
and category-category similarities.

Parameter Tuning. In our method, the parameter λ is designed to control the importance of user-item similarity and user-category similarity, whereas the parameter δ aims to control the depth of random walk on the multi-modal graph. To find the optimal value for these two parameters, we use the experimental setup similar to the previous sections, and plot the variation of averaged F1-score for each parameter. As shown in Fig 4, we empirically choose the optimal value $\lambda = 0.2$ and $\delta = 0.7$ for parameters.
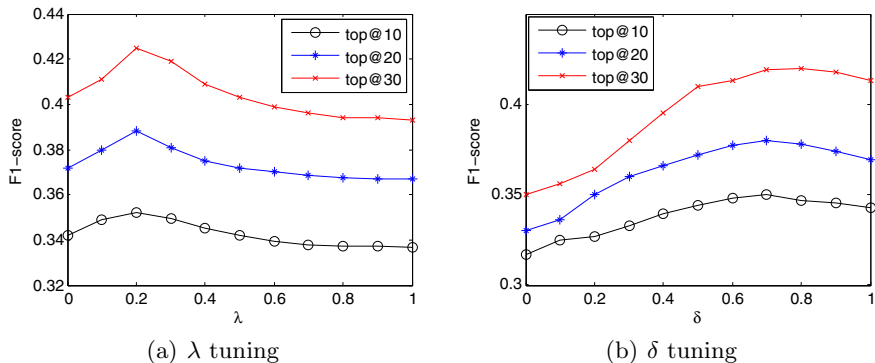

(a) λ tuning   (b) δ tuning

Fig. 4. Parameter tuning for λ and δ

6   Concluding Remarks

In this paper, we introduced a problem – Recommendation with Stage in E-commerce, and then proposed a novel method that utilizes item taxonomy to resolve this problem. Our approach is capable to effectively capture a customer's purchasing interest via a seamless integration of the customer's long-term profile and short-term profile, where the long-term profile is derived from the entire purchasing history, and the short-term profile is originated from the identification of the current recommendation stage of the customer. For recommendation purpose, we focus on the current recommendation stage and proposed a multi-modal graph ranking method to obtain the recommended item list. Evaluation on a dataset collected from *Amazon* demonstrated that the problem of recommendation with stage can be effectively tackled using our proposed method.

Notice that the performance of our proposed method depends on the quality of the item taxonomy. In our future work, we plan to investigate how the taxonomy would influence the performance of the proposed method. In addition, the segmentation of a user's purchasing history is based on a uniform time frame in our method, which is not quite sufficient since the durations of purchasing stages of different users might vary a lot. Therefore, we will also take into account different segmentation methods in our future work.

References

1. Cano, P., Koppenberger, M., Wack, N.: Content-based music audio recommendation. In: Proc. of ACM MM, pp. 211–212. ACM (2005)
2. Cantador, I., Bellog, A., Castells, P.: Ontology-based personalised and context-aware recommendations of news items. In: International Conference on Web Intelligence and Intelligent Agent Technology, pp. 562–565. IEEE (2008)
3. Chu, W., Park, S.: Personalized recommendation on dynamic content using predictive bilinear models. In: Proc. of WWW, pp. 691–700. ACM (2009)
4. Debnath, S., Ganguly, N., Mitra, P.: Feature weighting in content based recommendation system using social network analysis. In: Proceeding of the 17th International Conference on World Wide Web, pp. 1041–1042. ACM (2008)
5. Hung, L.: A personalized recommendation system based on product taxonomy for one-to-one marketing online. Expert systems with applications 29(2), 383–392 (2005)
6. Kang, J., Choi, J.: An ontology-based recommendation system using long-term and short-term preferences. In: International Conference on Information Science and Applications (ICISA), pp. 1–8. IEEE (2011)
7. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: Proc. of CIKM, pp. 247–254. ACM (2001)
8. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434. ACM (2008)
9. Lekakos, G., Caravelas, P.: A hybrid approach for movie recommendation. Multimedia Tools and Applications 36(1), 55–70 (2008)
10. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the 15th International Conference on Machine Learning, vol. 1, pp. 296–304 (1998)
11. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (2003)
12. Nakatsuji, M., Fujiwara, Y., Tanaka, A., Uchiyama, T., Fujimura, K., Ishida, T.: Classical music for rock fans?: novel recommendations for expanding user interests. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, pp. 949–958. ACM (2010)
13. Pan, J., Yang, H., Faloutsos, C., Duygulu, P.: Automatic multimedia cross-modal correlation discovery. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 653–658. ACM (2004)
14. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. Arxiv preprint cmp-lg/9511007 (1995)
15. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. Advances in Neural Information Processing Systems 20, 1257–1264 (2008)
16. Schafer, J., Konstan, J., Riedi, J.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM Conference on Electronic Commerce, pp. 158–166. ACM (1999)
17. Schickel-Zuber, V., Faltings, B.: Inferring user's preferences using ontologies. In: Proceedings of the National Conference on Artificial Intelligence, vol. 21, pp. 1413–1418 (2006)
18. Ziegler, C., Lausen, G., Schmidt-Thieme, L.: Taxonomy-driven computation of product recommendations. In: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 406–415. ACM (2004)

A GPU-Based Accelerator for Chinese Word Segmentation

Xiwu Gu[1,*], Ruixuan Li[1], Kunmei Wen[1], Bei Peng[1], and Weijun Xiao[2]

[1] Intelligent and Distributed Computing Lab, College of Computer Science and Technology
Huazhong University of Science and Technology, Wuhan 430074, P.R. China
`{guxiwu,rxli,kmwen}@mail.hust.edu.cn, beipeng.peng@gmail.com`
[2] Department of Electrical and Computer Engineering, University of Minnesota, twin cities
200 Union Street SE, Minneapolis, MN 55455, US
`wxiao@umn.edu`

Abstract. The task of Chinese word segmentation is to split sequence of Chinese characters into tokens so that the Chinese information can be more easily retrieved by web search engine. Due to the dramatic increase in the amount of Chinese literature in recent years, it becomes a big challenge for web search engines to analyze massive Chinese information in time. In this paper, we investigate a new approach to high-performance Chinese information processing. We propose a CPU-GPU collaboration model for Chinese word segmentation. In our novel model, a dictionary-based word segmentation approach is proposed to fit GPU architecture. Three basic word segmentation algorithms are applied to evaluate the performance of this model. In addition, we present several optimization strategies to fully exploit the potential computing power of GPU. Our experimental results show that our model can achieve significant performance speedups up to 3-fold compared with the implementations on CPU.

Keywords: Word Segmentation, GPU, CUDA.

1   Introduction

Chinese word segmentation is always the first task for Chinese information processing, and it is widely used in information extraction and information retrieval domain. As most web search engines locate the documents by keywords, the accuracy of word segmentation has a considerable effect on the quality of search results [1]. The objective of Chinese word segmentation aims to improve accuracy. However, Due to the explosive growth of Chinese literature online, Chinese information that needs to be processed by search engines becomes huge. The efficiency of segmentation thus becomes a prime factor affecting the performance of search engines [2]. Therefore, as an important part of information retrieval, word segmentation is expected to be more effective. To provide high throughput, search

---

* Corresponding author.

engines have to employ more CPUs to accelerate the processing of text analysis, index and search. In a distributed environment, the analysis tasks of Chinese texts are simultaneously launched on multiple CPU cores, and each core performs repetitive segmentation with different data set. As a result, the response time for processing massive data set is largely reduced. Though it is a straightforward solution to improve the performance of Chinese information processing by leveraging more computation resources, it will face a problem of increasing computing cost.

In this paper, we investigate a novel approach to enhancing the efficiency of Chinese word segmentation by using Graphic Processing Units (GPU). Generally, word segmentation is cutting a sentence into separate terms based on punctuation or similar rules. Each text is analyzed in the same way, so word segmentation can be regarded as a process sharing the instructions but with different data, which coincides with computing pattern of GPUs. Furthermore, segmentation on a large scale of texts will need huge repeated computation. Consequently, Chinese word segmentation becomes a potential application for GPU. GPU is designed for graphic rendering, and now it can perform a much higher degree of parallelism than CPUs, which offers many opportunities to exploit data parallelism. In recent years, lots of studies have attempted to use GPUs to accelerate the non-graphical applications, including scientific computing, analog emulation, and databases [3, 4, 5].

In our research, we introduce GPU to solve the performance bottle-neck problem of Chinese information processing for a large amount of data. We develop a CPU-GPU collaboration framework to analyze Chinese texts in parallel. Though GPU has provided a high-level programming interface, it is still a challenge task for developers to build a highly parallel data computation model. In addition, the utilization of streaming processors is another issue that will greatly affect the execution efficiency of GPU. In our framework, the GPU plays the role of a cooperator performing intensive, highly parallel computations rather than a rival to the CPU which is just responsible for scheduling and tasks distribution. By adding data buffer, the execution of word segmentation is launched asynchronously with text stream portability in a pipeline to overlap the I/O cost. In order to better suit to the GPU architecture, we design a parallel algorithm for the dictionary-based Chinese segmentation and optimize the instructions to delay memory access as much as possible. Furthermore, we design a cluster of data structures for GPUs and adopt several measures to cache the key data that is frequently read. Then, three Chinese word segmentation algorithms based on dictionary are performed on GPU to test the performance. We mainly evaluate the performance of this architecture in two aspects. First, we carry out a set of experiments to obtain the throughput in different situations, including parallel granularity and text size. The time cost in each phase of the task is also recorded. Then, our approach for parallel word segmentation is benchmarked against CPU, and several optimizations are carried out to improve the efficiency. The experiments indicate that our optimization strategy can well serve this new architecture, and GPU has advantages over CPU in data intensive computations.

The rest of the paper is organized as follows. We first review the related literatures of Chinese word segmentation and the studies of GPUs in different areas. Then, we introduce the CPU-GPU collaboration architecture and describe the implementation

of Chinese word segmentation on GPU. Moreover, some performance optimization technologies are integrated into this architecture for further acceleration. The experimental results are subsequently presented by the measurements of throughput and response time, and the paper concludes with a summary of parallel Chinese word segmentation on GPU.

2   Background and Related Work

2.1   Chinese Word Segmentation

Word segmentation is one of the most commonly researched areas in natural language processing (NLP), and it is always the basic work for information retrieval (IR). The texts should undergo a segmentation process to be broken up into smaller linguistic units before the words can be used to create index, only then texts can be searched by keywords. Therefore, word segmentation is such an important part that it has a great impact on the effectiveness of IR [1]. For languages like English, words are usually separated by space, and this explicit boundary is sufficient to divide a string of English into component words. However, some ideographic languages like Chinese do not mark a word in such a fashion, and it is a challenging task for computers to extract semantic words from a sentence automatically.

Chinese word segmentation has always been a very important research topic in Chinese text processing. Apart from Chinese information retrieval, it also plays a great role in other domains, such as Chinese input, speech recognition, machine translation and language understanding. In the past few years, most studies have looked into Chinese word segmentation for a better precision [6, 7]. The basic approaches involved in researches are mainly word-based, and they can roughly fall into two categories, namely the dictionary-based approach and the statistic-based approach. Most of the earlier work on Chinese word segmentation has concentrated on approach based on dictionary [8, 9], and it is commonly used in most current systems for coarse segmentation due to its simplicity and efficiency. For the dictionary-based approach, one of the most important components should be the dictionary which stores all possible words. When segmented, the chunk of strings can match against the dictionary to identify the words or phrases. However, when the dictionary becomes larger, it is difficult for CPU to locate a full word in serial mode. Therefore, the efficiency of searching dictionary turns to be a crucial aspect that influences the performance of word segmentation. To achieve quick prefix match, Fredkin [10] designed a digital search tree called Trie, whose time complexity is comparable to hash techniques. Aoe [11] simplified this prefix match tree into two parallel arrays for better space usage. The work in [12] introduced this structure to represent the dictionary for a high efficiency. The work in [13] designed a new structure to reduce space usage of the arrays. Nowadays, as the text set becomes larger, it is also a time-consuming job to process massive text data. So it is necessary to analyze the text sets in parallel besides improving dictionary structure. Multi-core device may be a good choice to improve the performance of word segmentation.

2.2   GPU

The graphic processing unit (GPU) is a dedicated device originally designed for rendering graphics. Compared to CPUs, modern GPUs are more efficient in manipulating computer graphics as a result of their highly parallel structure. Though GPU is a specialized circuit used for graphics processing, its high computing capability offers an opportunity to accelerate calculations for general-purpose computing. It is expected that GPUs can obtain a higher performance than CPUs when large blocks of data are processed in parallel. In the past few years, many engineers and scientists have increasingly exploited GPUs for non-graphical calculations, such as scientific computing and database [2, 3, 14]. Since NVIDIA launched the compute unified device architecture (CUDA), GPU becomes more programmable. In CUDA programming model, developers can transparently leverage the parallel engines on GPU with standard programming languages. Furthermore, data parallelism and task parallelism can be easily carried out with the help of thread groups. The job is partitioned into sub-tasks that are solved in parallel independently by blocks of threads, and then each sub-task is divided into finer pieces that are solved cooperatively by threads within the block [15]. Indeed, each block of threads is executed across the available processor cores concurrently or sequentially, and the kernel can be executed on any number of processor cores. So developers can just focus on the data flow rather than the traditional programming paradigm. Recently, the performance of modern GPU has increased at such a rapid pace that it is capable of outperforming current CPUs in some computation-intensive applications. More researchers have been exploiting GPU through CUDA for general-purpose computing [16, 17]. Several studies have been carried out to exploit GPUs to accelerate the computations involved in IR. The work in [18] addresses a problem of using GPUs to improve the efficiency of measuring document similarity. The work in [19] has implemented high-performance IR query processing on GPUs. They developed a set of parallel algorithms for index compression, index intersection, and ranking.

In the IR domain, indexing and searching always remain the focus as they are both the central parts for search engines. And recent works rarely present the issues on word segmentation because it seems to be in a secondary position. However, it is another case when search engines involve massive text processing. At that time, word segmentation will be a time-consuming process, and it will impose a big burden for CPUs. So it is desirable to use GPU to share the workload for search engines.

3   Implementation

GPU is specialized for intensive computations, so it can well address the problems that can be expressed as data-parallel. Large sets of data elements can be mapped to parallel threads, and each thread is executed for many data elements that have high arithmetic intensity. As a result, there is a lower requirement for sophisticated flow control. Many studies in non-graphic fields have adopted this data-parallel pattern to achieve better performance, such as physics simulation and computational finance. Word segmentation is also the one that fits the pattern. In general, the content that needs tokenizing is likely

to be a sentence, a paragraph or a piece of writing, and these individuals are almost isolated from each other anyway. By mapping independent text data to different threads, GPU can simultaneously launch all threads to start analyzing in parallel. However, it is not an easy task to perform parallel Chinese word segmentation on GPU. It requires frequent dictionary searching and big data transmission when the task is performed on GPU. All these movements are bound to generate large amounts of memory access operations. So, more optimizations are needed to make progress on GPU. Furthermore, many factors should be taken into account as a result of the limited resources on board. In this paper, we aim to solve these problems by designing the collaboration scheme and parallelism scheme. The collaboration scheme mainly coordinates the work between CPU and GPU, and further optimizes the data transmission. Parallelism scheme involves algorithm optimization to increase resources utilization. Both schemes play an important role in improving the overall performance of parallel word segmentation on GPU.

3.1   CPU-GPU Collaboration Architecture

The multi-core architecture of GPUs leads to a higher parallel computation capability than CPUs. However, the performance improvement would not be significant unless extra overhead is hidden. In this paper, we design CPU-GPU collaboration architecture to reduce the overhead and to make the best of GPU resource.
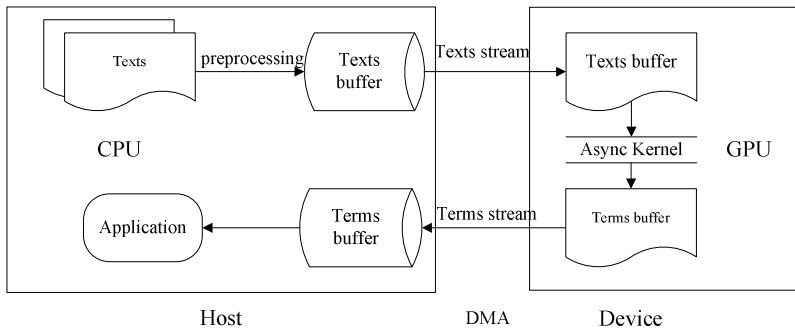

Fig. 1. CPU-GPU collaboration workflow

As shown in Fig. 1, the CPU is responsible for task scheduling while the calculation for word segmentation is offloaded to GPU. The task is carried out in batches which contain an amount of texts. First of all, raw texts are read from disk in batches incessantly and make a short stop in main memory. Before pushing into global memory, the batches should go through a pretreatment for load balance. On the host side, there is a text filter to convert Chinese texts to an appropriate format for GPU, and with the help of semantic separators, texts in a batch are divided into fragments with a full sense, so the workload can be controlled in fragments. Then the texts are pushed to texts buffer which maintains the texts to be analyzed by GPU. Once the device runs into idle time, the texts are immediately taken out from the texts buffer and transferred to global memory. In this case, CPU will be stuck at data

transmission phase in this case. For a better utilization of CPU, we asynchronously transfer data with the help of DMA. Furthermore, the data transmission task and calculation task are associated to work in a pipeline. As a result, streaming processors can keep running most of the time, and memory access could overlap parallel computation.

When the texts data is in place, each thread block is assigned one or more texts. To improve stream processor utilization, more tasks are distributed on GPU than there are enough cores. Then multiple threads on device side are launched concurrently to perform each task, and the terms extracted from the texts are pushed into terms buffer in a specified format. Finally, all terms are submitted in batches to host side asynchronously. In this cooperation model, GPU can continuously read data from texts buffer and put the results into the terms buffer without interference. In this case, streaming processors are always at full load, and some of the disk I/O time is hidden. Moreover, CPU could be extricated from the substantially repeatable computation with the assistance of GPU. The task for CPU is so limited that it can devote more to other applications. In a certain respect, this collaboration architecture can make better use of CPU and GPU to perform complex computations.

3.2   Parallel Segmentation on the GPU

Word segmentation based on dictionary is a typical task performed in serial. These operations during the segmentation procedure are bound to be closely linked with each other, so a finer-grained parallelism for dictionary-based segmentation is burdensome. Alternatively, we put the emphasis on data parallelism for a higher throughput. It has been proven that data parallelism can achieve a good performance on GPU. So data parallelism is supposed to be a promising way to increase efficiency of word segmentation. Though it seems a trivial task to achieve this kind of parallelism, it also takes a lot of efforts to perform an effective segmentation algorithm on GPU. For dictionary-based approach, dictionary usually contains lots of entries, which can occupy much space. Since the memory on GPU is limited, dictionary structure is of considerable importance. Besides, when the dictionary is extended, it will also cost a little more to find the right entry especially in global memory. So, parallel segmentation optimization is another important topic.

Dictionary construction faces a difficult choice. If a dictionary is small, many words will be missed during segmentation. Once the dictionary grows big, it will take up much valuable space. In our experiment, we select nearly 160 thousand Chinese words in common usage to make up the entries. In the dictionary, the percentage of two-word terms reaches half of the volume. Three-word terms and four-word terms share the rest with several terms exceeding ten words. Each character is hashed to a new value within two bytes so that dictionary space can be cut down. In general, linked list structure is flexible for dictionary update, but repeated node query in global memory may cause large latency due to the alignment problem. To better fit GPU architecture, all lexical information is compressed into two parallel arrays called Double Array [11]. As a result, the dictionary only takes up 3MB memory space and space utilization has exceeded 70%. Furthermore, we propose a state array to perform segmentation for different

demands in great ease. The extra array contains some auxiliary information such as transition information and plays a guidance role in word segmentation. Since the static dictionary no longer provides any update feature, we keep all the vocabulary information in files on host side. Then it is mapped to global memory when segmentation is required on device side. All dictionary data in global memory are aligned by 256 bytes in order that memory accesses within a warp can coalesce to one memory transaction. Even so, dictionary access is still a time-consuming task because the data for different threads are usually discontinuous in physical structure. To ease the pressure of memory access, we make use of the texture memory for data cache. In this case, some of the vocabulary information is located in texture memory. So it is unnecessary to access global memory every time when matching the dictionary. For segmentation in some specified field, it is likely to put this tiny dictionary in shared memory, which can largely reduce the memory access time.

The task of parallel word segmentation on GPU is illustrated in Fig. 2. The texts are first assigned to blocks, and then each block distributes the calculations across threads. When all texts data for each block is in place, all threads on device side are lunched concurrently to perform text analyzing. Each thread independently checks the character against dictionary one by one. When the transition information is not hit in texture memory, it then goes forward to the global memory. Once a term occurs, the thread put it into the terms buffer owned by each block. Since there is no synchronization overhead among threads in a block, this data-parallel pattern can achieve a good performance of word segmentation on GPU.
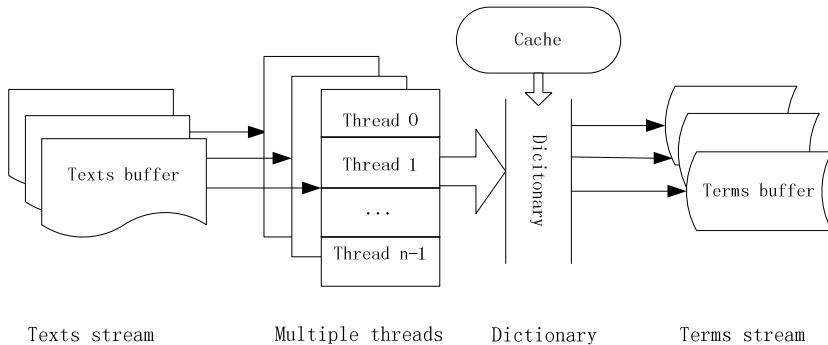

Fig. 2. Parallel word segmentation model

When the segmentation task is complete, another kernel is launched to count the frequency of each term. In order to keep load balancing, each block receives a block of terms and assigns a sub-block of equal size to each thread. Then each thread compares the terms to the current global term in sequence and changes the frequency value in frequency table. When all threads in a block complete the comparisons, the cursor of global terms moves to the next. Though this approach can achieve frequency counting for each term, it is inefficient on GPU as the threads should be synchronized in a block. Moreover, since the terms obtained by the segmentation kernel may be repeated in the memory, it will take extra time to remove the duplicate terms.

Therefore, we propose another approach to fulfill these two tasks. In this approach, the task of frequency counting is performed before the word segmentation task. The frequency of each term is immediately recorded when the term is extracted from the piece of text. In order to reduce the synchronizing operation among threads, we designed a global hash table for term mapping and a frequency table for each block based on the dictionary structure. Moreover, we use the final transition state of a term to identify the location in frequency table. During the segmentation phase, each thread only changes the frequency value in frequency table. When all threads finish the segmentation task, a frequency table has been already complete for each block. So it is straightforward for each thread to identify the terms that occur in the text by checking the frequency table. Then all threads can get the terms values from the hash table to fill the terms buffer. The improved algorithm executed by each thread is shown in Algorithm 1.

Algorithm 1. Kernel for word segmentation and frequency count

```
    Input: Texts set, Dictionary
    Output: Terms and frequency set
 1  while more texts in texts buffer
 2    string • get_piece(tid)
 3    for each character c in string
 4      s • match_dictionary_texture(c)
 5      if a term appears then
 6        block_freq(s) increases
 7      else
 8        Move to next character
 9      end if
10    end for
11  end while
12  Locate terms in hash_table based on block_freq
13  Write terms to terms buffer
```


In this way, we first count the frequency of each term, and then obtain the terms of each text. As a result, all terms can be easily obtained in one time with the help of the hash table and frequency table, and each term is recorded only once in global memory. Furthermore, it leads to a decrease in the number of memory access as it is unnecessary to access the memory each time a term occurs during the segmentation phase. Moreover, it won't take too much synchronization overhead to perform the frequency statistics task in a block as the task of frequency counting is achieved by changing the frequency table instead of counting the duplicate terms. So it is more efficient for GPU to perform parallel word segmentation and term frequency statistics in this manner.

4   Experimental Results

During the experiment, the results obtained on GPU are benchmarked against CPU, and more experiments are carried out to evaluate the performance of this collaboration model. Our experiments are carried out on windows server 2003 with a

dual-core Intel CPU at a speed of 2.00GHz. The type of GPU is GeForce GTS 250 which has 128 CUDA cores. The Chinese texts are selected from Chinese law and regulation document set.
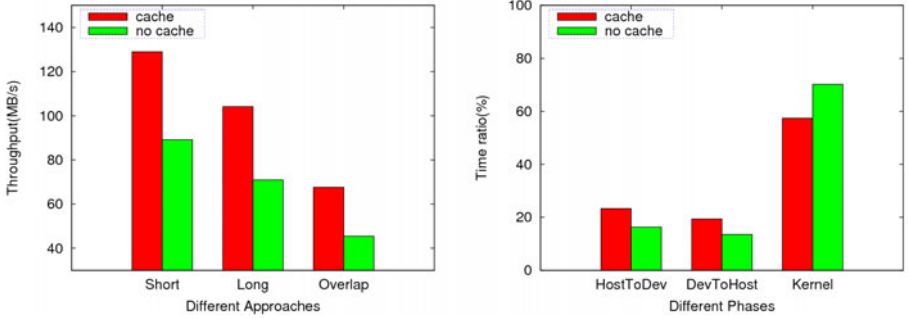
(a) Throughput of GPU with cache

(b) Time ratio in different phases

Fig. 3. Performance of GPU with cache

For word segmentation, dictionary is an important resource that is accessed frequently. As it will suffer a great latency for each thread to access global memory, we propose to put some of the dictionary information into the cache of texture memory. So it is unnecessary for threads to match the dictionary in global memory each time. It is shown in Fig. 3(a) that the throughput is increased greatly with the help of cache. In Fig. 3(b), we can find that the time shared by the kernel is reduced while the time of I/O transmission is relatively increased. It is demonstrated that less time is cost to access global memory during segmentation when the dictionary is cached in texture memory. Therefore, stream processors can perform more tasks per unit time so that the overall time has been reduced.
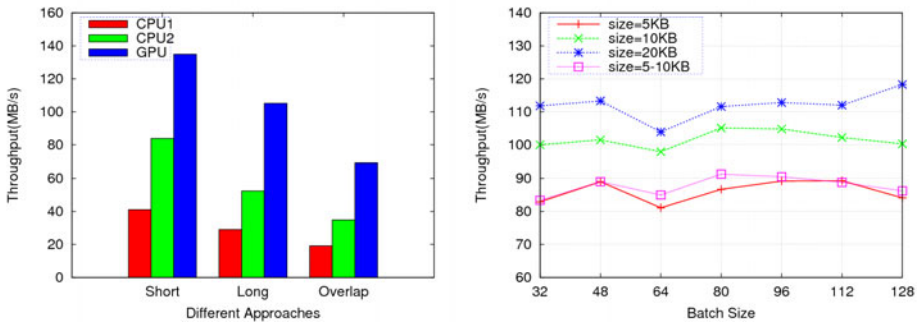
(a) Throughput of CPU and GPU

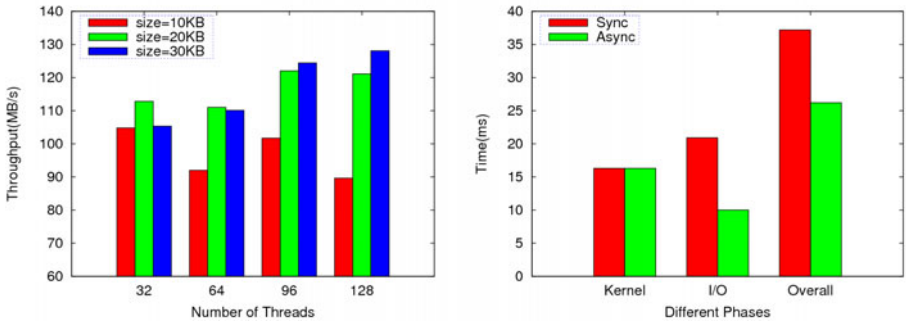(b) Throughput in different text sizes

Fig. 4. Performance of GPU in different approaches and text sizes

In order to evaluate the performance of GPU, three Chinese word segmentation approaches are performed on CPU and GPU respectively. It is depicted in Fig. 4(a) that the throughput of GPU is two times more than that of single-core CPU and

almost twice as much as that of dual-core CPU. It can be inferred that GPU can reach the calculation capability of quad-core CPU in parallel segmentation. Moreover, since the shortest match approach usually divides a string into two-word terms, most of the threads can run in the same path with less thread falling into different branches. So the shortest match approach can achieve the maximum throughput while the longest match approach falls behind.

Furthermore, we carry out experiments on longest match approach to study the impact of task granularity on the efficiency of streaming processors. It is depicted in Fig. 4(b) that the throughput becomes higher as the text size of each batch gets larger. So, more calculations can improve the utilization of each stream processor before it is full-load. However, it is not the case when the size of texts in the same batch is different. Though the total size of batch grows the throughput also stays at a low level due to the unbalance load across the stream processors.

Besides load balance, parallelism granularity is an important aspect to improve the performance of GPU. It is depicted in Fig. 5(a) that more threads can better solve the problem of big data processing. So it can achieve a better performance to assign more threads to share the load when the text size grows. However, it will cause a side effect if the text is too small. Further study reveals that it has nothing to do with the workload assigned to each thread. In other words, heavier load for each thread may not necessarily cause higher throughput. It is the scheduling for multiple threads that influences the performance of stream processors. When the text size is rather too small, the overhead for managing additional threads exceeds the cost to perform the task. As a result, the resource on device side can't be fully utilized. However, once the text size gets larger, more threads are needed to share the workload so that stream processors are at full load. So, the throughput eventually grows in this case.

(a) Throughput in different threads counts   (b) Time cost in different modes

Fig. 5. Throughput and response time of GPU

Since GPU is a coprocessor, the I/O overhead should be taken into account. More experiments are carried to study the time cost of different phases in parallel segmentation. It is shown in Fig. 5(b) that the I/O overhead contributes significantly to the overall time for a batch in synchronous fashion, and the I/O time unexpectedly exceeds the kernel time, which cannot be endured in efficiency. When the task of word segmentation is performed in asynchronous model, CPU can be free from data

transmission between main memory and global memory with the help of DMA. Meanwhile, stream processors are always running at full load on device side. As a result, the I/O time is cut by half in this case as the kernel execution overlaps the transmission operations.

The performance of GPU is further evaluated when it is used for both word segmentation and frequency statistics. It is shown in Table 1 that the statistics kernel costs most of the time to deal with a batch about 1MB, but the segmentation kernel is finished just in a while. The GPU puts up a poor performance in the task of frequency statistics, so we choose to count the term frequency before recording terms. With the help of a hash table and block wise frequency tables, the efficiency is largely enhanced. It is depicted in Table 1 that the response time of GPU is greatly reduced by combining word segmentation and frequency statistics into one kernel. Since the combined kernel consumes an amount of time, the time for I/O transmission becomes insignificant in this case. It is shown in Table 1 that the time for the task about 1GB in asynchronous mode is less than that of synchronous mode as the time for I/O transmission is well hidden by kernel execution.

Table 1. Execution time of GPU in different phases and modes

Approach	Time(ms)	Execution mode	Time(s)
Segmentation kernel	7.1	Sync mode	126.5
Statistics kernel	491.9	Async mode	118.7
Combined kernel	117.3		

5   Conclusions

In this paper, we present an implementation of parallel Chinese word segmentation using GPU. We design the CPU-GPU collaboration architecture and implement the segmentation algorithm. To exploit the computation capability of GPU, we optimize the task granularity and load balance from different perspectives. Though various inherent branches of segmentation algorithms have influenced the execution efficiency of GPU, we still achieve a 3-fold speed-up. It is well demonstrated that our approach is effective in parallel Chinese word segmentation, and GPU can be exploited to accelerate massive data processing.

References

1. Foo, S., Li, H.: Chinese word segmentation and its effect on information retrieval. J. Inf. Process. Manage. 40(1), 161–190 (2004)
2. Thompson, C.J., Hahn, S., Oskin, M.: Using modern graphics architectures for general-purpose computing: a framework and analysis. In: 35th International Symposium on Microarchitecture, pp. 306–317. IEEE Press, New York (2002)

3. Govindaraju, N.K., Lloyd, B., Wang, W., Lin, M.C., Manocha, D.: Fast computation of database operations using graphics processors. In: SIGMOD, pp. 215–226. ACM Press, New York (2004)

4. Preis, T., Virnau, P., Paul, W., Schneider, J.J.: GPU accelerated Monte Carlo simulation of the 2D and 3D Ising model. J. Comput. Phys. 228(12), 4468–4477 (2009)

5. Fan, Z., Qiu, F., Kaufman, A.E., Yoakum-Stover, S.: GPU cluster for high performance computing. In: SC, p. 47 (2004)

6. Gao, J.F., Li, M., Wu, A., Huang, C.N.: Chinese word segmentation and named entity recognition: a pragmatic approach. J. Computational Linguistics 31, 574 (2005)

7. Wang, X.J., Qin, Y., Liu, W.: A search-based Chinese word segmentation method. In: 16th International World Wide Web Conference, pp. 1129–1130. ACM Press, New York (2007)

8. Yang, W., Ren, L.Y., Tang, R.: A dictionary mechanism for Chinese word segmentation based on the finite automata. In: International Conference on Asian Language Processing, pp. 39–42 (2010)

9. Liu, X.G., Luo, J., Xie, Z.: The Research of Chinese Automatic Word Segmentation In Hierarchical Model Dictionary Binary Tree. In: First International Workshop on Database Technology and Applications, pp. 321–324. IEEE Press, New York (2009)

10. Fredkin, E.: Trie memory. J. CACM 3(9), 490–499 (1960)

11. Aoe, J.I.: An efficient digital search algorithm by using a double-array structure. J. IEEE Trans. Software Eng. 15(9), 1066–1077 (1989)

12. Zheng, C., Zheng, Q.H., Zhou, Z., Tian, F.: A method for large cross-language lexicon management based on collaborative work of hash family and double-array trie. In: 14th International Conference on Computer Supported Cooperative Work in Design, pp. 658–663. IEEE Press, New York (2010)

13. Yata, S., Oono, M., Morita, K., Fuketa, M., Sumitomo, T., Aoe, J.I.: A compact static double-array keeping character codes. J. Inf. Process. Manage. 43(1), 237–247 (2007)

14. Krüger, J., Westermann, R.: Linear algebra operators for GPU implementation of numerical algorithms. J.ACM Trans. Graph. 22(3), 908–916 (2003)

15. NVIDIA CUDA Programming Guide (2010),
    `http://developer.download.nvidia.com/`

16. Ryoo, S., Rodrigues, C.I., Baghsorkhi, S.S., Stone, S.S., Kirk, D.B., Hwu, W.M.W.: Optimization principles and application performance evaluation of a multithreaded GPU using CUDA. In: 13th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, pp. 73–82. ACM Press, New York (2008)

17. He, B.S., Yang, K., Fang, R., Lu, M., Govindaraju, N.K., Luo, Q., Sander, P.V.: Relational joins on graphics processors. In: SIGMOD, pp. 511–524. ACM Press, New York (2008)

18. Zhang, Y.P., Mueller, F., Cui, X.H., Potok, T.: GPU-accelerated text mining. In: Workshop on Exploiting Parallelism using GPUs and other Hardware-Assisted Methods. ACM Press, New York (2009)

19. Ding, S., He, J.R., Yan, H., Suel, T.: Using graphics processors for high performance IR query processing. In: 18th International World Wide Web Conference, pp. 421–430. ACM Press, New York (2009)

The Equi-Join Processing and Optimization on Ring Architecture Key/Value Database

Xite Wang, Derong Shen, Tiezheng Nie, Yue Kou, and Ge Yu

Key Laboratory of Medical Image Computing (NEU), Ministry of Education
College of Information Science and Engineering, Northeastern University, P.R. China
xite-skywalker@163.com,
{shenderong,nietiezheng,kouyue,yuge}@ise.neu.edu.cn

Abstract. Huge data analysis (e.g. equi-join) on key/value data storage system is a very novel and necessary issue. For the master-slave architecture distributed system(MSADS) (e.g. Hadoop), equi-join can be implemented by utilizing MapReduce framework which is designed for performing scalable parallel data analysis on MSADS. However, for the ring architecture distributed system (RADS), there has no general method for processing data analysis task (e.g. equi-join). Hence in this paper, a novel approach is proposed for processing equi-join on RADS. Firstly, by making in-depth analysis of RADS, we propose a new type of index, ColumnValue index(CVI), based on ColumnFamily data model. Then, by utilizing CVI, an efficient algorithm, called pre-join table generation algorithm (PJTG), is proposed to process the equi-join query on RADS, and a memory index(MI) is utilized to further improve the performance of PJTG. In addition, the update method for the join result is present , and the update is caused by the alteration of original data. Finally, the validity of PJTG is verified through plenty of simulation experiments. Experimental results show that the proposed algorithm is an effective way to solve the equi-join query problem on RADS and could meet the requirements of practical applications.

1 Introduction

In the age of information explosion, very large data sets pose a new challenge for the data storage systems. Companies (e.g. Google) need to analyze petabytes of information from web pages. Facebook wants to find the relationship among hundreds of millions of people. The traditional relational database has been unable to meet the requirements. To achieve high reliability and scalability, many data storage systems have adopted the distributed architecture. According to the difference of the architectures, the distributed data storage systems can be classified into two categories. One is the master-slave architecture distributed system (e.g. Hadoop [1]). This kind of system is designed for the data warehouse which has poor performance in data random access. The other kind is ring architecture distributed system which has higher availability. This kind of system is appropriate for storing the data which often updates (e.g. web information), for instance, Cassandra [2] in Facebook and Dynamo [3] in Amazon.

For many data analysis tasks, join query especially the equi-join is very essential. The MapReduce [4] framework is highly desirable for performing scalable parallel data analysis on MSADS and equi-join can be implemented by utilizing MapReduce [5,6]. Recently, some open source projects (e.g. Pig [7] and Hive [8]) have already realized this function. Moreover, some more in-depth research has been done, for instance, multi-way join [9] and inequality-join [10] based on the MapReduce framework. However, according to our survey, most of the research does not consider the situation in which the original data updates frequently.

Because there is no mater node and each node in the cluster is parallel, RADS cannot support the MapReduce framework directly. For the data analysis tasks, there is no universal paradigm on RADS. For the equi-join process, the developed algorithm on MSADS using MapReduce framework can't be utilized on RADS as well. This issue poses a new challenge for the data analysis on RADS.

In this paper we propose a novel algorithm that enable efficient parallel execution of equi-join on RADS without MapReduce. Furthermore , we design the update method on RADS. In particular, we make the following contributions.

1. Based on the ColumnFamily data model, we propose CVI on ColumnValue, and utilize the consistency hash algorithm to store CVI on the cluster.
2. In order to further improve efficiency, we build MI, and use MI to reduce the useless disk access.
3. By utilizing CVI and MI, a new algorithm, called Pre-join Table Generation(PJTG) algorithm is proposed to process the equi-join task on RADS.
4. For the alteration of the join result caused by the original data updating, we also design the method to maintain the exactness of the join result table and the data structure we propose.

The rest of the paper is organized as follows. Section 2 briefly introduces the related knowledge and our devise goals. Section 3 discusses the details of our proposed PJTG algorithm. Section 4 explains the update approach of join result as the extension of PJTG. Section 5 describes the experimental results and related analysis. Finally, the conclusion of this paper is stated in Section 6.

2   Preliminaries

2.1   Overview of Distributed Architecture

As we explained earlier, there are two types of distributed systems, MSADS and RADS. For MSADS, master node is the single director of the cluster and it stores the meta-data of the entire system, and each slave node is only responsible for the local data storage. For RADS, there is no mater node, each node in the cluster is parallel. RADS utilizes the consistent hashing algorithm [11] to organize and manage the cluster. Figure 1 illustrates the architecture of MSADS and RADS.

As Figure 1(a) shows, for MSADS, while the MapReduce task starts, the JobTracker on the master node m becomes the task controller, it is responsible for the task allocation and maintains the balance of workload on each slave
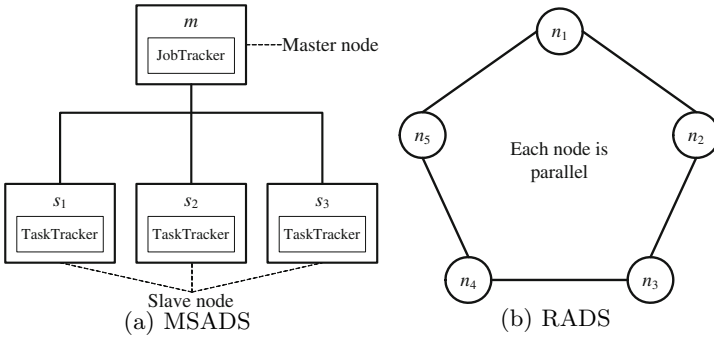
Fig. 1. The architecture of distributed systems

node. Slave nodes (e.g. s_1, s_2) become the workers on which the subtask is executed, and there is no communication between slave nodes. As we have seen, in a sense, the master node is the key of MapReduce. In Figure 1(b), there is no mater node, each node is parallel in RADS. All the nodes maintain the route information of the entire cluster. Therefore, RADS has high availability without single point of failure. However, unfortunately, it is unable to process the traditional MapReduce task directly on RADS.

2.2 ColumnFamily Data Model

ColumnFamily data model comes from BigTable [12]. A ColumnFamily is equivalent to a Table in RDBMS. Figure 2 illustrates an instance of ColumnFamily.

RowKey	Columns		
	ColumnKey	ColumnValue	Timestamp
k_1	c_1	v_1	t
	c_2	v_2	t
k_2	c_1	v_3	t
	c_2	v_4	t

Fig. 2. ColumnFamily data model

As figure 2 shows, ColumnFamily contains a plenty of rows and each row is a key/value pair. The key of a row is defined as RowKey (e.g. k_1) and it is one and only in order to distinguish other rows. The value of a row is a set of columns. A column is a key/value pair with a timestamp. We define the key of column as ColumnKey (e.g. c_1) and the value as ColumnValue (e.g. v_1). Note that there is no independent index on column because each data can be obtained only by utilizing the RowKey in the key/value data storage systems . In order to facilitate the expression, we use Row(k) to represent the row whose RowKey is k , Col(c) to represent the column whose ColumnKey is c.

2.3   Join Conditions Based on ColumnFamily

According to the definition above, the condition of equi-join based on ColumnFamily data model can be redefined into three types. For $Col(c_1)$ in ColumnFamily cf_1 and $Col(c_2)$ in cf_2, the three types of conditions are:

$$cf_1.RowKey = cf_2.RowKey \tag{1}$$

Condition 1 means the join condition of two ColumnFamilies is that the Rowkey of cf_1 is equal to the Rowkey of cf_2.

$$cf_1.c_1.ColumnValue = cf_2.RowKey \tag{2}$$

Condition 2 means the join condition is the ColumnValue of $Col(c_1)$ in cf_1 is equal to the Rowkey of cf_2.

$$cf_1.c_1.ColumnValue = cf_2.c_2.ColumnValue \tag{3}$$

Condition 3 means the join condition is the ColumnValue of $Col(c_1)$ in cf_1 is equal to the ColumnValue of $Col(c_2)$ in cf_2. Obviously, the third type is the most representative and the other two are the simplified models from it. Therefore, in this paper, we mainly study on the third type of equi-join.

2.4   Optimization Goals

Unlike the MapReduce framework, there is no control centre while executing data analysis task on RADS. Therefore, the approach we design must be able to be executed on each node of the cluster independently which has low mutual constraint and high parallel processing capability. Moreover, as we mentioned in Section 1, the data stored on RADS updates frequently. Hence, while the original data changes after the equi-join task achieves, the join result must update to keep validity. The situation of update needs to be considered in our algorithm.

3   PJTG Algorithm

The PJTG contains three phases. Firstly, two CVIs for the join-condition columns should be built. Secondly, find the rows with the same RowKey in both CVIs, and MI is built to avoid the useless disk read. Finally, accomplish the euqi-join and generate the join result. The detail is as follows.

3.1   Build CVI

For the equi-join condition 3, firstly, CVI, which is similar to the secondary index [13], should be built for each join-condition column(e.g. c_1, c_2) of the ColumnFamily . In detail, CVI is a ColumnFamily structure logically, and it is designed to search the RowKey for a given ColumnValue of a specified column. Figure 3 illustrates an instance of CVI.
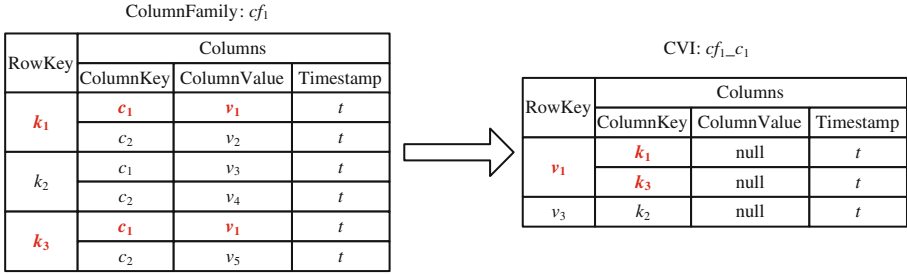
ColumnFamily: cf_1

RowKey	Columns		
	ColumnKey	ColumnValue	Timestamp
k_1	c_1	v_1	t
	c_2	v_2	t
k_2	c_1	v_3	t
	c_2	v_4	t
k_3	c_1	v_1	t
	c_2	v_5	t

CVI: $cf_1\_c_1$

RowKey	Columns		
	ColumnKey	ColumnValue	Timestamp
v_1	k_1	null	t
	k_3	null	t
v_3	k_2	null	t

Fig. 3. Example of CVI

As Figure 3 shows, CVI $cf_1\_c_1$ is built for $Col(c_1)$ in ColumnFamily cf_1. In the CVI, RowKeys are the ColumnValues of $Col(c_1)$ in cf_1. For a $Row(v_1)$ in CVI, ColumnKeys(e.g. k_1, k_3) are the RowKeys whose ColumnValue of $Col(c_1)$ in cf_1 is v_1, and the ColumnValue is *null*. Via CVI $cf_1\_c_1$, for a given ColumnValue v_1 of $Col(c_1)$ in cf_1, we can get all the RowKeys whose ColumnValue of $Col(c_1)$ is v_1 and all the information of those rows in cf_1.

For the storage of CVI, the consistent hashing algorithm is employed. Assume CVI for $Col(c)$ in ColumnFamily cf needs to be built, for a given hash function h and a data storage node i, for each $Row(k)$ in cf on node i, the ColumnValue v of $Col(c)$ could be obtained easily. Via $h(v)$, the address p where the $Row(v)$ of CVI cf_c stores could also be computed. Finally, the insert message is sent to the node whose address is p.

Each node of the cluster could performance the above period of process independently. Hence, there is no need for the control centre to keep parallel. In addition, the MD5 random hash function is utilized to maintain the load balance. According to our simulation experiment, workload on each node is almost the same. For the join condition 3, CVI $cf_1\_c_1, cf_2\_c_2$ should be built according to the approach above. The mission of next phase is to find the rows with the same RowKey in CVI $cf_1\_c_1, cf_2\_col_2$ and get the set of ColumnKeys of each row. Note that owe to the consistent hash distributed storage type of the CVI, the rows with the same RowKey must be stored at the same node. Hence, the second phase can also be executed on each node independently without the communications between nodes.

3.2 Build Memory Index

To achieve the mission of the second phase, at local address of each node, the general method is that for each RowKey k which is get from CVI $cf_1\_c_1$(this operation contains one disk read), find the Row whose RowKey is equal to k in CVI $cf_2\_c_2$(this operation contains another disk read). Maybe there is no such row in CVI $cf_2\_c_2$, but we still have to perform two disk read operations at least to make sure that. This series of operations especially the disk read are time-consuming and useless. This situation is called *MissRead*. In some practical applications, the MissRead is very frequent and causes the obvious performance degradation. Therefore, this case must be considered in our algorithm.

To avoid MissRead, an index maintained in the memory, MI, is built on each node independently. In detail, MI is a hashmap. The key of MI is the RowKey of both CVI $cf_1\_c_1$ and $cf_2\_c_2$ at local address. The value of a key k should be small enough to enable mass index, but the pointer which points to the disk position of $\mathrm{Row}(k)$ is not appropriate. Because in RADS, the storage position of a row is adjusted sometimes(e.g. the data compaction). In that case, a plenty of the pointers will change, which causes the MI updates in a large scale. Hence the way making the pointer as the value of MI is unseemly. In our algorithm, for a key k of MI, the value is a array with two items, and each item is a counter which flags the amount of columns in $\mathrm{Row}(k)$ of one CVI. For instance, assume that the first item of the array is the counter for CVI $cf_1\_c_1$ and second is for $cf_2\_c_2$, a insert message $\mathrm{SET}(cf_1\_c_1, v, k, null)$ is received at node i, which means insert a data into CVI $cf_1\_c_1$ with RowKey v, ColumnKey k, ColumnValue $null$. Then browse MI to get the value $flags$ of key v, and make $flags[0]$ plus 1. Note that for a key k in MI, when neither the two items of the value are equal to 0, the $\mathrm{Row}(k)$ must be the one which cannot cause MissRead, so we can store k into a set V. Finally, the set V is all the exact RowKeys we need.

Obviously, when the system finishes building the CVI, the MI is accomplished too. Moreover, join-condition columns (e.g. c_1, c_2) is either the major key or the foreign key in a ColumnFamily commonly. Therefore the schema of the Column-Family can be confirmed while creating, and all the phases above can be executed as pretreatment before join operation. The pretreatment will greatly reduce the time cost of the whole process.

3.3 Accomplish the Join Process

In the final phase, firstly for each v in the set V generated in the second phase, get the ColumnKey set ck_1 which belongs to the $\mathrm{Row}(v)$ in CVI $cf_1\_c_1$, and ck_2 in $cf_2\_c_2$ with the same way. Make cartesian product of ck_1 and ck_2 to get the result set $keys$. According to each ordered pair in $keys$, we can return to cf_1 and cf_2 to extract the rows we need, and insert them to the join result ColumnFamily $ResultCF$. The RowKey is named as $k_1\_k_2$ if a $\mathrm{Row}(k_1\_k_2)$ in $ResultCF$ is the join result of $\mathrm{Row}(k_1)$ in cf_1 and $\mathrm{Row}(k_2)$ in cf_2.

3.4 PJTG Description

Algorithm 1 describes the flow path of the whole process. Firstly, some necessary variable is initialized(Line 1). Secondly, call function createIndex(cf, c, la, i)(See Algorithm 2 and 3 for detail) to generate CVI, MI, and the RowKey set of CVI without MissRead V(Line 2 - 3). Then, for each Rowkey v in set V, get the ColumnKey sets of both CVI and make cartesian product to generate $decareList$(Line 4 - 8). Finally for each ordered pair $\langle k_1, k_2 \rangle$ in $decareList$, return to the ColumnFamily cf_1, cf_2 and get the data we need, then insert it into the join result(Line 9 - 13). Note that we utilize the predicate pre_1, pre_2 to extract the exact columns we want and reduce the size of join result.

Algorithm 1. PJTG algorithm

---

 input : ColumnFamily name cf_1, cf_2; condition ColumnKey c_1, c_2; total
              number of node n; local address la

 output: the join result ColumnFamily $resultCF$

1 hashmap $h= \emptyset$; set $V= \emptyset$; int $i= 0$;
2 createIndex(cf_1, c_1, la, i++);
3 createIndex(cf_2, c_2, la, i);
4 **for** *each v in set V* **do**
5   list *decareList= $\emptyset$*;
6   string[ ] $firstKeys$ = localGet($cf_1\_c_1$, v).getColumnKeys();
7   string[ ] $secondKeys$ = localGet($cf_2\_c_2$, v).getColumnKeys();
8   *decareList*= cartesianProduct ($firstKeys$, $secondKeys$);
9   **for** *each ordered pair $\langle k_1, k_2 \rangle$ in decareList* **do**
10   list *columnsList= $\emptyset$*;
11   *columnsList*= get(cf_1, k_1, pre_1).getColumns();
12   *columnsList*.addAll(get(cf_2, k_2, pre_2).getColumns());
13   insert($resultCF$, $k_1\_k_2$, *columnsList*);

14 **end**;

Function createIndex(cf, c, la, i) mentioned in Algorithm 1(Line 2 - 3) is used
to create CVI, MI, and the RowKey set V, which contains two parallel threads
a for the main data process at local address(described in Algorithm 2) and b for
monitoring the message from other nodes(described in Algorithm 3).

Algorithm 2. createIndex(cf, c, la, i) (thread a)

---

 input : total number of node n; local address la

 output: CVI cf_c; hashmap h

1 CVI $cf\_c= \emptyset$; int $count= 0$;
2 **for** *each RowKey k of cf at local address* **do**
3   string $v=$ localGet(cf, k, c).getColumnValue();
4   string $ad=$ getAddress(v);
5   **if** $la == ad$ **then**
6   localInsert(cf_c, v, k, $null$);
7   MIInsert(h, v, i);
8   **else**
9   sendInsertMessage(ad, cf_c, v, k, $null$);

10 sendCompleteMessage();
11 $count$++;
12 **end**;

In Algorithm 2, Firstly for each Row(k) of cf at local address, get the Colum-
nValue v of Col(c)(Line 2 - 3) and compute the storage node address ad of v
as RowKey of CVI cf_c(Line 4). If ad is equal to local address la, insert this
data into CVI cf_c locally and call function MIInsert(h, v, i)(See Algorithm 4
for detail)(Line 5 -7). Else, send the insert message to the node with address

ad(Line 8 - 9). At last, broadcast the message over the cluster to indicate the accomplishment and make *count* plus 1(Line 10 - 11).

In Algorithm 3, a message m is received from the listen port(Line 1). If m is a insert message, insert this data into CVI cf_c locally and call function MIInsert(h, v, i)(Line 2 - 4). If m is an accomplishment message, add 1 to *count*(Line 5 - 6). When *count* is equal to the total number n, which means all the nodes in the cluster finish this period of process, the algorithm ends.

Algorithm 3. createIndex(cf, c, la, i) (thread b)

    input : total number of node n; local address la
    output: CVI cf_c; hashmap h

1 monitor message port, get message m;
2 **if** $m.type==$ ”*insert*” **then**
3   localInsert($cf\_c, v, k, null$);
4   MIInsert(h, v, i);
5 **if** $m.type==$ ”*complete*” **then**
6   *count*++;
7 **if** *count* $== n$ **then**
8   end;

Algorithm 4 describes the function MIInsert(h, v, i) which arise in Algorithm 2(Line 7) and 3(Line 4). Firstly, get the value *temp* of the key v in the hashmap(Line 1). If *temp* is an empty set, initialize an integer array *flags* and turn *flags*[i] to 1, then insert pair(v, *flags*) into h(line 2 - 5). If *temp* is not an empty set and *temp*[i] is 0, add 1 to *temp*[i]. Note that this is the first time neither item of *temp* is 0, which means RowKey v is not able to cause MissRead, so insert v into V(Line 6 - 9). If *temp* is not an empty set and *temp*[i] is not 0, add 1 to *temp*[i] only(Line 10 - 11).

Algorithm 4. MIInsert(h, v, i)

    input : hashmap h; RowKey in CVI v
    output: RowKey set without MIssRead V

1 int[ ] *temp*= h.get(v);
2 **if** *temp*$== \emptyset$ **then**
3   int[ ] *flags*= {0; 0};
4   *flags*[i]++;
5 h.set(v, *flags*);
6 **if** *temp* $\neq \emptyset$ **then**
7   **if** *temp*[i]$== 0$ **then**
8   *temp*[i]++;
9 V.add(v);
10   **else**
11   *temp*[i]++ ;
12 end;

4   Extension of PJTG

As we mentioned in the introduction, the data stored in RADS updates frequently. After the equi-join achieves, the data structures we build such as CVI, MI and the join result ColumnFamily need to keep update when the original data changes. Hence, in this section, we will introduce the approach for update. As a prerequisite condition, for condition 3, the result ColumnFamily $ResultCF$ for ColumnFamily cf_1, cf_2 has been built by utilizing PJTG algorithm.

The approach for the update caused by the join-condition column contains two types of operations, DELETE and INSERT. In addition, INSERT contains INC and MOD operation. The detail is described as follow.

For DELETE(cf_1, k_1, c_1) which means the join condition column, Col(c_1) of Row(k_1) in cf_1, is asked to be deleted, before this operation is done, the original ColumnValue v of Row(k_1), Col(c_1) in cf_1 must be obtained. Then on the node where the Row(v) of CVI $cf_1\_c_1$ stores, CVI $cf_1\_c_1$ and MI need update. Then get the ColumnKey set $keys$ of Row(v) in CVI $cf_2\_c_2$. For each k in $keys$, delete Row(k_1_k) in $ResultCF$.

For INSERT(cf_1, k_1, c_1, v_1) which means a ColumnValue v_1 is asked to insert into Row(k_1), Col(c_1) in cf_1, first of all, the original ColumnValue v of Row(k_1), Col(c_1) in cf_1 must be obtained. If v is $\emptyset$, which means this is a new write operation, this operation can be transformed to INC(cf_1, k_1, c_1, v_1). In this case, on the node where the Row(v_1) of CVI $cf_1\_c_1$ stores, CVI $cf_1\_c_1$ and MI need update. Then get the ColumnKey set $keys$ of Row(v) in CVI $cf_2\_c_2$. For each k in $keys$, new Row(k_1_k) joined by Row(k_1) in cf_1 and Row(k) in cf_2 is wrote into $ResultCF$.

If v is not $\emptyset$, which means this is a modification operation, this operation can be transformed to MOD(cf_1, k_1, c_1, v_1). In this case, before INSERT done, DELETE(cf_1, k_1, c_1) needs to be executed to delete the overdue data. Then INC(cf_1, k_1, c_1, v_1) is executed to add new data while the original data updates.

The approach for the update caused by the none-join-condition column is straightforward because there is no alterations of CVI and MI. For DELETE(cf_1, k_1, c_0) which means the none-join-condition column, Col(c_0) of Row(k_1) in cf_1, is asked to be deleted. Firstly, ColumnValue v of Row(k_1), Col(c_1)(the join-condition column) in cf_1 needs to be obtained. Then on the node where the Row(v) of CVI $cf_1\_c_1$ stores, get the ColumnKey set $keys$ of Row(v) in CVI $cf_2\_c_2$. For each k in $keys$, delete Col(c_0) of Row(k_1_k) in $ResultCF$. The INSERT operation is similar to DELETE, then we omit the details.

5   Experimental Evaluation

In this section, we will evaluate the performance of our proposed PJTG approach by simulation experiment with JAVA programming language. In detail, our PJTG algorithm is implemented based on the open source RADS Cassandra and a part of the code is modified for actual requirement. All the experiments are executed on the cluster built by 20 no-shared personal computers with Intel

Core i7 870 CPU, 8G DDR3 memory, 500GB hard disk. In contrast, we process equi-join utilizing MapReduce framework on MSADS Hadoop. The cluster is built by 20 data nodes and 1 name node with the same hardware configuration.

We use runtime to evaluate the efficiency of our PJTG algorithm. The co-efficient of variation of the data size of join result on each node C_D and the coefficient of variation of the runtime of process on each node C_T is employed to evaluate the expansibility and the degree of parallelism of PJTG. The experiment data sets are synthetic. In both of the original ColumnFamilies, each row has 50 columns and we extract 10 columns to compose the join result Column-Family. The runtime of PJTG at the rate of MissRead 33% is showed in Figure 4(a), and 50% in Figure 4(b).
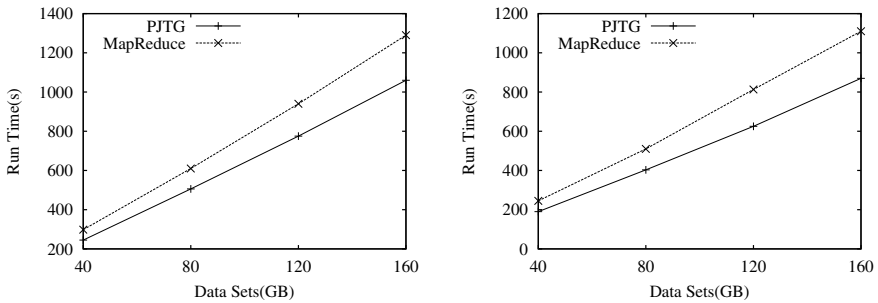
(a) Runtime at the rate of MissRead 33% (b) Runtime at the rate of MissRead 50%

Fig. 4. Runtime of PJTG

Figure 4(a) shows the runtime of equi-join with different data size of original ColumnFamilies at the rate of Missread 33%. The abscissa represents data size and the ordinate represents runtime. The experiment result shows that our PJTG is very efficiency and the relationship between data size and runtime is almost linear. In contrast, the same equi-join is executed on Hadoop with the same original data sets. As we see, equi-join using our PJTG on RADS is nearly 20% faster than using MapReduce framework on MASDS. This increase is mainly because that there is no need of communication between slave node and master node, and MissRead is avoided in our PJTG algorithm.

Figure 4(b) shows the runtime of equi-join at the rate of MissRead 50%. The solid line represents equi-join using our PJTG and the dotted line represents the same query on Hadoop. Our PJTG on RADS is nearly 28% faster than MapReduce. Hence, the experiment shows that with the increase of MissRead, our PJTG shows higher performance.

In Figure 5, the data size of the join result on each node is showed. In detail, the size of original data sets is 160GB and contain two ColumnFamilies. Both the ColumnFamilies contain 40 millions rows. The join result ColumnFamily is nearly 64GB and contain 80 millions rows. The rate of MisstRead is 33%. As the Figure 5 shows, Owe to the distributed hash algorithm, $C_D = 0.00229$, which means the data distribution on each node is very balanced.
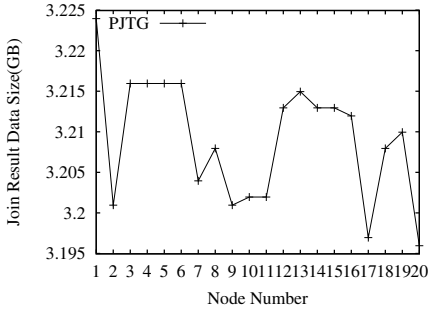
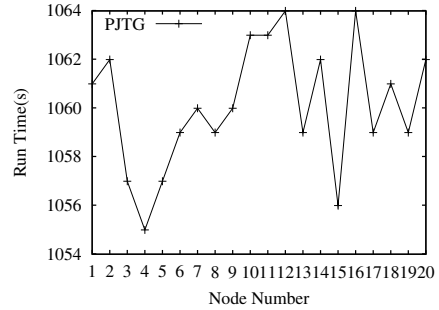**Fig. 5.** Data size on each node of the cluster

Fig. 6. Runtime on each node of the cluster

Figure 6 shows the runtime of our approach on each node with the same experiment data sets in Figure 5. $C_T = 0.00238$, which means the process on each node almost achieves at the same time. The consequence of Figure 5 and Figure 6 illustrates that PJTG has very high expansibility and degree of parallelism.

Finally, for the update of the join result ColumnFamily caused by the alteration of original data, we also measure the runtime of update operations. The size of join result ColumnFamily is 64GB and each update time is the average time of 10000 same operations. Table 1 shows the update time of DELETE, INSERT(INC) and INSERT(MOD) operations on join-condition column and DELETE and INSERT operations on non-join-condition column. As Table 1 shows, the update operation is very fast and can meet the practical requirements. In addition, commonly, operations on the join-condition column is slower than that on the non-join-condition column because that while we modify the data on join condition column, there will be some time cost for the alteration on CVI and MI.

Table 1. The update of join result

	join-condition column			non-join-condition column	
	Delete	Insert(INC)	Insert(MOD)	Delete	Insert
Time cost(ms)	4.3	4.1	4.7	4	4
Alteration on CVI and MI	Y	Y	Y	N	N

Through the experiments above, we can conclude that the PJTG algorithm has high expansibility and degree of parallelism. It is very efficiency and shows high performance while the situation of MissRead arise frequently. the time cost of update is also very low, which makes our approach more practical.

6   Conclusions

Data analysis task especially equi-join over huge data sets stored on Key/Value database is very necessary in many applications. MapReduce is a appropriate

framework to solve this issue on MSADS, but there is no general approach for equi-join on RADS. Therefore, in this paper, we focus on the parallel execution of equi-join on RADS. Through in-depth analysis of RADS, CVI and MI is proposed firstly. Then based on the data structure we propose, a novel approach, called PJTG, is proposed to performance equi-join task on RADS efficiently. Moreover, the update method for the join result is also designed as the extension of PJTG. Finally, a large number of experiments verify that our proposed PJTG is an efficient algorithm with high parallelism and availability, and can meet the practical requirements. There are still some limitations of the study such as system flexibility and fault tolerance. Next, we will focus on these problems, and design a more general framework of parallel process on RADS. In order to further improve the performance, we will also design the physics storage unit on RADS for our framework.

References

1. Apache hadoop, `http://hadoop.apache.org`
2. Avinash, L., Prashant, M.: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review 44(2) (April 2010)
3. Giuseppe, D., Deniz, H., Madan, J.: Dynamo: Amazon's Highly Available Key-value Store. In: SOSP, Washington, USA, October 14-17, pp. 205–220 (2007)
4. Jeffrey, D., Sanjay, G.: MapReduce: simplified data processing on large clusters. In: OSDI, California, USA, December 6-8 (2004)
5. Foto, A., Jeffrey, U.: Optimizing Joins in a Map-Reduce Environment. In: EDBT, Lausanne, Switzerland, March 22-26 (2010)
6. Kamil, P., Daniel, A., Avi, S.: Efficient Processing of Data Warehousing Queries in a Split Execution Environment. In: SIGMOD, Athens, Greece, June 12-16, pp. 1165–1176 (2011)
7. Apache pig, `http://pig.apache.org/`
8. Apache hive, `http://hadoop.apache.org/hive`
9. Yuting, L., Divyakant, A., Chun, C.: Llama: Leveraging Columnar Storage for Scalable Join Processing in the MapReduce Framework. In: SIGMOD, Athens, Greece, June 12-16, pp. 961–972 (2011)
10. Alper, O., Mirek, R.: Processing Theta-Joins using MapReduce. In: SIGMOD, Athens, Greece, June 12-16, pp. 949–960 (2011)
11. Karger, D., Lehman, E., Leighton, T.: Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spotson the World Wide Web. In: STOC, USA, pp. 654–663 (1997)
12. Fay, C., Jeffrey, D., Sanjay, G.: Bigtable: A Distributed Storage System for Structured Data. In: OSDI, Seattle, WA, USA, November 6-8 (2006)
13. Apache cassandra, `http://cassandra.apache.org/`

Efficient Probabilistic Image Retrieval Based on a Mixed Feature Model[*]

Yi Zhuang[1], Zhiang Wu[2], Nan Jiang[3], Guochang Jiang[4],
Dickson K.W. Chiu[5], and Hua Hu[6]

[1] College of Computer and Information Engineering, Zhejiang Gongshang University, P.R. China
[2] Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, P.R. China
[3] Hangzhou No.1 People's Hospital, Hangzhou, P.R. China
[4] The Second Institute of Oceanography, SOA, Hangzhou, P.R. China
[5] Dickson Computer Systems, HKSAR, P.R. China
[6] School of Computer, Hangzhou Dianzi University, P.R. China
`zhuang@zjgsu.edu.cn`

Abstract. In content-based image retrieval (CBIR) methods, color, texture, and shape are extracted from an image as low-level visual features for searching. However, these visual features could not express image sentiment concepts required in applications like garment search. To address this issue, we propose an efficient image retrieval method based on combined visual and conceptual feature spaces. A unified similarity distance between two images is obtained by linearly concatenating similarity measures. To further improve the retrieval efficiency, we propose an efficient probabilistic indexing scheme called _Mixed-Feature-based Probabilistic Tree_(MFP-Tree) to facilitate the retrieval over a large image repository. Different from conventional image retrieval and indexing methods, which only adopt visual similarity as a query metric, our proposed retrieval algorithm allows users to choose among the above three kinds of features as query elements. Moreover, a probabilistic model is introduced to refine the retrieval result with confidence guarantee. Comprehensive experiments have testified the effectiveness and efficiency of our proposed retrieval and indexing method.

Keywords: high-dimensional indexing, probabilistic retrieval, sentiment.

1   Introduction

Content-based image retrieval (CBIR) has been extensively studied due to its wide application in many fields. Typical methods [1][2] extract color, texture, and shape

---

from an image as low-level visual features for retrieval, but their effectiveness (e.g., *recall* and *precision*) is often unsatisfactory due to the semantic gap between high-level concepts and low-level visual features [2]. According to recent cognition research, besides widely adopted visual features, some conceptual ones such as *style*, and even *sentiment* of an image can affect the retrieval accuracy to some extent, which have not been considered into traditional similarity-based search over image databases. Fig. 1 shows an example garment retrieval application, in which two different images with similar visual features (e.g., *color*) may be subjectively identified by users as two different styles such as '*Vogue*' and '*Elegant*'. Moreover, in Figs. 2(a) and 2(b), two different sentiments, '*Happy*' and '*Sad*', are conveyed in two images, respectively. Often, users would like to retrieve some images with a specific style or sentiment they prefer to, which cannot be achieved by traditional visual feature-based search. Therefore, we propose a mixed- feature-based image retrieval method that combines visual and conceptual features together to facilitate image retrieval.

| (a). Vogue | (b). Elegant | (a). Happy | (b). Sad |

Fig. 1. Under two different styles   **Fig. 2.** Under two different sentiments

Table 1. The probabilistic & confidence distribution of the styles

Style	Probability	Confidence
Vogue	60%	70%
Japanese-like	25%	60%
Elegant	5%	65%
Vogue and Japanese-like	10%	75%

Table 2. The probabilistic & confidence distribution of the sentiments

Sentiment	Probability	Confidence
Happy	5%	75%
Angry	5%	65%
Melancholy	75%	60%
Sad	10%	70%
Encouraging	5%	60%

   In fact, due to the content complexity of an image, personal knowledge level, and errors occurred in the feature extraction, as shown in Tables 1-2, identifying the corresponding conceptual features of an image (e.g., *image style* and *sentiment*) are often uncertain and non- trivial. For example, for the image in Table 1, the probabilities of the image style are '*Vogue*', '*Elegant*', '*Japanese-like*', and '*Vogue*

and Japanese-like' are 60%, 25%, 5%, and 10%; while user-provided probabilistic confidences are 70%, 60%, 65%, and 75%, respectively. Given another example image in Table 2, due to the user's mood, the probabilities that the image whose sentiments belong to '*Happy'*, '*Angry'*, '*Melancholy'*, '*Sad'*, and '*Encouraging'* are 5%, 5%, 75%, 10%, and 5%, with confidences 75%, 65%, 60%, 70%, and 60%, respectively. Therefore, it is necessary to introduce a more sophisticated probabilistic model to represent the conceptual features of an image effectively.

After addressing the retrieval effectiveness issue by modeling images with both visual and conceptual features, the next key issue is the efficiency, which directly relates to the problem of high-dimensional data indexing with multiple features. Although considerable research efforts have been done on high-dimensional indexing issues [5], unfortunately, existing indexing methods cannot be directly applied to our combined feature-based images retrieval due to the existence of the probabilistic data.

To address this issue, we propose a probabilistic and mixed high-dimensional indexing scheme based on multiple features, called MFP-Tree. With the aid of the MFP-Tree index, the issue of a probabilistic k nearest neighbor(PkNN) query of image λq in a high-dimensional space can be solved by transforming it into a range query in the single dimensional space. The primary contributions of this paper are as follows:

1. We present an effective retrieval method by choosing combined features (i.e., *visual and conceptual features*) of image, in which a *multi-reference-based probability propaga- tion*(MRPP) method is introduced to obtain a feature probability and confidence distribution table in the preliminary step of the retrieval.

2. We introduce a <u>Mixed-Feature Probabilistic-Tree</u>(MFP-Tree)-based indexing method to facilitate the interactive and efficient images retrieval with multiple features.

3. We perform extensive experiments to evaluate the effectiveness and efficiency of our proposed probabilistic retrieval method and indexing scheme.

The remainder of this paper is organized as follows. Section 2 introduces some background of our work. Section 3 presents our preprocessing steps of our method. In Section 4, we propose a <u>Mixed Feature-based Probabilistic Tree</u>(MFP-Tree)-based high-dimensional index- ing scheme to improve the retrieval efficiency dramatically. Section 5 reports our results of extensive experiments to evaluate the efficiency and effectiveness of our approach, before we conclude in the final section.

2   Background

Image retrieval research has been extensively studied during the past two decades. QBIC [1] was known as the first system to support content-based image retrieval. After that, many prototype systems have been built, such as VIRAGE [2], Photobook [3], and MARS [4]. In these CBIR systems, however, only objective low-level visual features such as color histogram, texture, and shapes are adopted, without considering subjective features such as style, sentiment. For example, in garment information

systems, users often want to retrieve images that are not only visually similar, but also satisfying user intentions in some subjective ways, e.g., *style* and *sentiment*, etc. In this paper, we further study an effective image retrieval method by extending the traditional CBIR techniques from a perspective of feature subjectivity.

Multiple feature indexing issues belong to multi(*high*)-dimensional indexing category [5]. The R-tree [6] and its variants [7] are based on data & space partitioning, hierarchical tree index structure. However, their performance becomes unsatisfactory as dimensionality increases [5]. Another approach is to represent original feature vectors using smaller, approxi- mate representations such as VA-file [8] and IQ-tree [9]. Alternatively, iDistance [10] is a distance-based approach. Recently, Jagadish *et al.* [11] have devised an indexing method to facilitate the similar retrieval using multiple features. Worse still, these indexing approaches are only suitable for indexing multi-dimensional data without probabilistic components. Feature uncertainty is often neglected, which is testified to be inherent in many real applications including feature extraction from images and so on. Few researches have addressed indexing multiple features with uncertainty components so far.

3   Preprocessing Steps of Our Method

3.1   Modeling an Image

First we briefly introduce the notations to be used in the rest of this paper. For an image, we model two kinds of features, namely, visual features and conceptual ones. For conceptual features of an image, there exists a number of different styles and sentiments, respectively (see Figs. 1 and 2). Thus the image I_i can be represented by a four-tuple:

$$I_i ::= <i,\ Vfea,\ Sty,\ Senti > \tag{1}$$

where

- i refers to the i-th image in Ω;
- *Vfea* refers to the visual features of the image such as color histogram, texture, and shape, etc;
- $Sty=\{StyID,\ StyName,\ StyVal,\ P_{ST},\ C_{ST}\ \}$, where *StyID* is the style ID of *Ii; StyName* is the style name of *Ii, StyVal* is the style value of *Ii such as 'vogue', 'elegant', 'classic', P_{ST}=* **Prob**(the style ID of *Ii* is *StyID*), C_{ST} is the average user confidence of the style selection.
- $Senti=\{SenID,\ SenName,\ SenVal,\ P_{SE},\ C_{SE}\ \}$, where *SenID* is the sentiment ID of *Ii*, *SenName* is the sentiment name of *Ii*, *SenVal* is the sentiment value of *Ii such as 'happy', 'sad', 'encouraging', P_{SE}* refers to the average probability that the sentiment ID of *Ii* is *SenID*), where formally, P_{SE}=**Prob**(SenID(*Ii*)=*SenID*), C_{SE} is the average user confidence of the sentiment;

To fully utilize these two features to effectively prune the search space, we propose a probabilistic retrieval scheme in which the style and sentiment of each image can be identified by users in the preprocessing step.

Table 3. Meaning of Symbols Used

Symbols	Notations	Symbols	Notations
Ω	a set of images	Iq	a query image user submits
I_i	the i-th image and $I_i \in \Omega$	$\Theta(Iq,r)$	a query sphere with centre Iq and radius r
n	the number of images in Ω	I_R	Reference image
m	the number of reference images	I_q	a query image user submits
$Sim(I_i,I_j)$	the unified similarity distance between two images	$stSim(I_i,I_j)$	the style similarity distance between two images
$vSim(I_i,I_j)$	the visual similarity distance between two images	$seSim(I_i,I_j)$	the sentiment similarity distance between two images

Definition 1 (Visual vector)**.** *A visual vector (VIV) of an image I_i is a vector, denoted as $VIV_i = <VI_{i1}, VI_{i2} ,…, VI_{id}>$, where ST_{ij} refers to the value of the i-th bin of the j-th visual vector, $i \in [1,n]$ and $j \in [1,d]$; d is the dimension of the VIV.*

Definition 2 (Style vector)**.** *A style vector (STV) of an image I_i is a vector, denoted as $SV_i = <ST_{i1}, ST_{i2} ,…, ST_{id'}>$, where $ST_{ij}=(styname_{ij}, prob_{ij})$, $i \in [1,n]$, and $j \in [1,d']$; styname$_{ij}$ can be 'vogue', 'elegant', and 'classics', etc; d' is the dimension of the STV.*

For example, for the i-th image, the probability distribution of the different style is <60%, 25%, 5%, 10%>. For the j-th image, the probability distribution of the different style is <50%, 30%, 10%, 10%>. The style similarity distance(*stSim*) is calculated below.

$$stSim(\lambda_i,\lambda_j) = \sqrt{(60\% - 50\%)^2 + (25\% - 30\%)^2 + (5\% - 10\%)^2 + (10\% - 10\%)^2} \approx 0.2449$$

Definition 3 (Sentiment vector)**.** *A sentiment vector(SEV) of an image I_i is a vector, denoted as $SEH_i=<SE_{i1}, SE_{i2} ,…, SE_{id''}>$, where $SE_{ij} =(sentname_{ij}, prob_{ij})$, $i \in [1,n]$, and $j \in [1, d'']$; sentname$_{ij}$ can be 'happy', 'angry', and 'melancholy', etc; d'' is the dimension of the SEV.*

For the i-th image, suppose that the probabilities that its sentiments belongs to '*happy*', '*angry*', '*melancholy'* and '*sad'* are 70%, 15%, 5%, 10%, respectively. And for the *j*-th image, the probability distribution of the different sentiments is <50%, 30%, 10%, 10%>. So the sentiment similarity distance (*seSim*) of the two images is as follows.

$$seSim(\lambda_i,\lambda_j) = \sqrt{(70\% - 50\%)^2 + (15\% - 30\%)^2 + (5\% - 10\%)^2 + (10\% - 10\%)^2} \approx 0.2549$$

3.2 Building a Probabilistic Model

3.2.1 Learning from User's Log

As a first step of the preprocessing, suppose there are m reference images, for each reference image I_{Ri}, its corresponding styles and sentiments can be selected by t users with probabilistic confidences, thus I_{Ri} can be represented by a five-tuple:

$$I_{Ri} ::= <i, j, fName, fVal, Confi_{ij}> \tag{2}$$

where i refers to the image ID of I_{R_i} in Ω and $i\in[1,m]$, j means the user ID and $j\in[1,t]$, *fName* means the conceptual features such as style or sentiment, *fVal* is the feature value, $Confi_{ij}$ is the probabilistic confidence with which the *j*-th user choose one of the conceptual features (i.e., *fName*) of the image *IRi*. The following table shows an example of user's log in which *id* refers to the image ID and *Uid* is the user ID.

Table 4. An example of user log table

Id	Uid	fName	fVal	$Confi_{ij}$
i	j	style	'Vogue'	60%
i	j+1	style	'Vogue'	70%
i	j+2	style	'Vogue'	80%

Suppose t users are involved in the selection of the conceptual features. Then, the average probabilistic confidence($Conf_X$) can be derived as:

$$Conf_X(I_{R_i}) = \frac{1}{t}\sum_{j=1}^{t} Confi_{ij} \tag{3}$$

where X can be ST (i.e., style), or SE (i.e., sentiment).

3.2.2   Feature Probability and Confidence Distribution Table

As mentioned before, for each image I_i in Ω, its corresponding probabilistic distributions of different features (such as *visual features*, *style,* and *sentiment*) with some confidence guarantees are important to facilitate the image retrieval with the combined features. Therefore, as a preprocessing step in the retrieval process, it is critical to build a feature probabilistic distribution table for each image, which is defined as follows.

Definition 4 (Feature Probability & Confidence Distribution Table, *FPCDT*). *A feature probability distribution table of the i-th image, denoted as $FPCDT_i$, consists of five parts:*

$$FPCDT_i := <i,\ Fname,\ Fval,\ Prob,\ Conf_X> \tag{4}$$

where i refers to the image ID, Fname is the feature name (such as style and sentiment), Fval means the corresponding feature value, Prob is the probability that the feature value equals to Fval, and $Conf_X$ means the average user confidence of the conceptual feature (X can be style or sentiment).

Table 5 shows an example of the *FPCDT* in which the probabilities that the style and sentiment of the 1st image equals to different feature values(*Fval*) are illustrated with confidence(*$Conf_X$*) guarantee.

For n images in Ω, it is a labor-intensive work to obtain the n *FPCDTs*. So in the pre-processing step, in order to obtain the *FPCDT* of each image, we propose a *multi-reference- based probability propagation*(MRPP) method, in which m images are first randomly chosen as reference ones and $m<<n$. The corresponding *FPCDTs* of

Table 5. An example of *FPCDT*

ID	i	Fname	Fval	Prob	Conf$_X$
1	1	Style	'Vogue'	60%	70%
2	1	Style	'Elegant'	20%	60%
3	1	Style	'Classics'	30%	65%
4	1	Style	'Japanese-like'	10%	70%
5	1	Sentiment	'Happy'	20%	75%
6	1	Sentiment	'Angry'	10%	70%
7	1	Sentiment	'Sad'	70%	80%
8	1	Sentiment	'Encouraging'	0%	70%

these reference images are first manually obtained through the interactions of users. For the other (*n-m*) images, their corresponding *FPCDTs* can be derived via the affected reference images of them with respect to the visual feature distance.

Definition 5 (Affected Reference Image Region, *ARIR*)**.** *Given an image I$_i$, its affected reference image region, denoted as ARIR(I$_i$), is a hypersphere centered with I$_i$ and radius θ in the visual feature space.*

Definition 6 (Affected Reference Image, *I$_R$*)**.** *Given an image I$_i$, its j-th affected reference images, denoted as I$_{Rj}$(i), are the reference images occurred in the corresponding ARIR in terms of visual similarity, where j∈[1, δ] and δ refers to the number of affected reference images.*
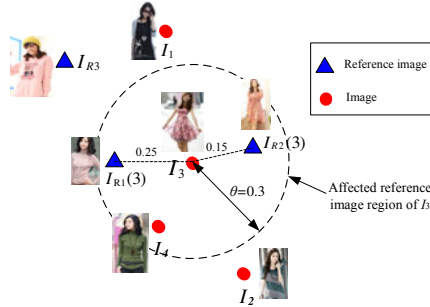
Fig. 3. An example of the multi-reference-based probability propagation

As mentioned before, once *m FPCDTs* are constructed, the next step is to derive *FPCDTs* of other (*n-m*) images, the probability that the feature value of the *i*-th image equals to *Y* can be derived as:

$$\mathrm{Prob}\big(\text{The image feature}(X)\text{ of } I_i = Y\big)$$
$$=\sum_{j=1}^{\delta}\left(\lambda_j \times \mathrm{Prob}\big(\text{The image feature}(X)\text{ of } I_{Rj}(i) = Y\big) \times \frac{vSim(I_{Rj}(i), I_i)}{\sqrt{2}} \right) \quad (5)$$

where

— X can be style or sentiment, etc.;
— If X refers to style, then Y can be '*Vogue*', '*Elegant*', '*Classics*' and '*Japanese-like*', etc. If X refers to sentiment, then Y can be '*Happy*', '*Angry*', '*Sad*' and '*Encouraging*', etc.;

- λ_j is a tuning parameter and $\sum_{j=1}^{\delta}\lambda_j = 1$; and
- δ refers to the number of affected reference images.

Similarly, the confidence that the feature value of I_i equals to Y can also be propagated as follows:

$$
\begin{aligned}
&Conf_X\left(\text{The image feature}(X)\text{ of } I_i = Y\right)\\
&=\frac{\sum_{j=1}^{\delta}\left[Conf_x\left(\text{The image feature}(X)\text{ of } I_{Rj}(i)=Y\right)\times vSim(I_{Rj}(i),I_i)\right]}{\sum_{j=1}^{\delta}Conf_x\left(\text{The image feature}(X)\text{ of } I_{Rj}(i)=Y\right)}
\end{aligned}
\tag{6}
$$

where the symbols (e.g., X, Y, and δ) are defined the same as in Eq.(5).

For example, in Fig. 3, suppose that there are seven images such as I_1, I_2, I_3, I_4, I_{R1}, I_{R2}, and I_{R3} in which three images (e.g., I_{R1}, I_{R2} and I_{R3}) are considered as reference ones. For image I_3, given a radius θ, the affected reference images fall in the *ARIR* are I_{R1} and I_{R2}. So $\delta=2$.

Suppose $vSim(I_3,I_{R1}(3))=0.25$, $vSim(I_3,I_{R2}(3))=0.15$, $\theta=0.3$ and $d=32$, according to Eqs.(5) and (6), then we have:

Prob(The style feature value of I_3 = '*Vogue*')

$$
\begin{aligned}
&=\sum_{i=1}^{2}\left(\lambda_i\times Prob\left(\text{The style feature of } I_{Ri}(3)=Y\right)\times\frac{vSim(I_{Ri}(3),I_3)}{\sqrt{2}}\right)\\
&=\frac{1}{2}\times0.05\times\frac{0.25}{\sqrt{2}}+\frac{1}{2}\times0.1\times\frac{0.15}{\sqrt{2}}\approx0.0097=0.97\%
\end{aligned}
$$

where $\lambda_1=\lambda_2=1/2$.

Similarly, the probabilities that the style feature value of I_3 equals to '*Elegant*', '*Classics*', and '*Japanese-like*' are 1.9%, 10%, and 1.1%, respectively. Since 0.97%+1.9%+10%+1.1% = 13.97%<100%, so we need to conduct a uniform process of the probability values. The uniform probability values are shown in Fig.4(c).

For the image I_3, the confidence that its corresponding style feature value is '*Vogue*' can be derived as follows:

$$
\begin{aligned}
&Conf_{ST}\left(\text{The style feature value of } I_3 = '\text{Vogue}'\right)\\
&=\frac{\sum_{j=1}^{\delta}\left[Conf_x\left(\text{The style feature of } I_{Rj}(3)='\text{Vogue}'\right)\times vSim(I_{Rj}(3),I_i)\right]}{\sum_{j=1}^{\delta}Conf_x\left(\text{The style feature of } I_{Rj}(3)='\text{Vogue}'\right)}\\
&=\frac{0.7\times0.25+0.8\times0.15}{0.25+0.15}=73.8\%
\end{aligned}
$$

Analogously, the confidences that the style names of I_3 belong to '*Elegant*', '*Classics*', and '*Japanese-like*' are 11.6%, 15.9%, and 10.6% accordingly, which are shown in Fig.4(c).

Style	Prob.	Confi.
Vogue	5%	70%
Elegant	15%	80%
Classics	70%	90%
Japanese-like	10%	65%

(a). Probability & confidence distribution of $I_{R1}(3)$

➜

Style	Prob.	Confi.
Vogue	6.8%	73.8%
Elegant	13.4%	81.9%
Classics	71.7%	90%
Japanese-like	8.1%	61.3%

(c). Probability & confidence distribution of I_3

Style	Prob.	Confi.
Vogue	10%	80%
Elegant	10%	85%
Classics	75%	90%
Japanese-like	5%	55%

(b). Probability & confidence distribution of $I_{R2}(3)$

Fig. 4. The construction of the FPCDT of image I_3 via the two reference ones $I_{R1}(3)$ and $I_{R2}(3)$

Algorithm 2. The MRPP-based *FPCDT* construction algorithm
Input: *m* reference images λ_q, *StyID or SenID*
Output: the probability distribution table of the *n* images
1. $S \leftarrow \Phi, S1 \leftarrow \Phi$;   //   *initialization*
2. **for** each image I_i **do**
3.   choose the affected reference images $I_{Rj}(i)$ in its corresponding *ARIR*;
4.   the average probability that the feature(X) of I_i belongs to Y can be derived by Eq.(5);
5.   the average confidence users hold can be obtained in Eq.(6);
6. **return** *FPCDT*;   //   *return the probability distribution table*

Fig. 5. The MRPP-based *FPCDT* construction algorithm

4   A MFP-Tree-Based Probabilistic Retrieval Algorithm

In this section, we propose an indexing method called the MFP-Tree (*Mixed-Feature-based Probabilistic Tree*) to facilitate an image retrieval method with probabilistic confidence guarantee.

4.1   Motivation and Preliminaries

As a preliminary step, before constructing the MFP-Tree, all images in Ω are first grouped into *T* clusters by *k*-Mean clustering algorithm in terms of their visual features. Each cluster is denoted as C_j, where $j \in [1,T]$. So we can model a cluster as a tightly bounded sphere described by its *centroid* and *radius*, which is saved in a class information file.

Definition 7 (Cluster Radius). *Given a cluster C_j, the distance between O_j and the image which has the longest distance to O_j is defined as the cluster radius of C_j, denoted as CR_j.*

Definition 8 (Centroid Image). *Given an image I_i and its corresponding cluster C_j, the centroid image of I_i is the centroid object O_j in C_j, denoted as CR_j, where $I_i \in C_j$ and $O_j \in C_j$.*

Given a cluster C_j, its corresponding cluster sphere can be denoted as $\Theta(O_j, CR_j)$, where O_j is the centroid image of cluster C_j, CR_j is the cluster radius.

Once T clusters are obtained, then the distance between each image and its corresponding centroid one is computed. Moreover, its style and the sentiment are identified at the same time. Finally, a uniform index key of an image is obtained to be discussed in Section 4.3.

4.2   Determining Weights by Multivariable Regression

Note that when we construct the mixed feature vectors using linear concatenation from the visual features (e.g., *color histogram*, *texture*, etc), style, and sentiment feature vectors, we cannot assume that each type of visual and conceptual features contributes equally in image recognition for human cognition, as different image feature plays different roles for effective image retrieval.

Some earlier research work on image retrieval and the evaluation of image similarity that underlies it offers different weight specifications. In this work, we introduce a scheme to determine the weight for each feature based on multivariable regression. In the model, the distance between two image items, *Sim*, can be presented as a linear function of distance for each visual and conceptual features. Symbolically, it can be written as below,

$$Sim(I_i,I_j)=w_v\times vSim(I_i,I_j)+w_s\times stSim(I_i,I_j)+w_e\times seSim(I_i,I_j) \tag{7}$$

where $w=[w_v,w_s,w_e]$ is the vector of weight coeffcients to be determined (v, s, and e denote visual feature, style, and sentiment feature, respectively) and $d=[vSim(I_i,I_j), stSim(I_i,I_j), seSim(I_i,I_j)]$ is the vector of independent distance value for each type of feature.

4.3   The Data Structure

In order to effectively prune the search region, we propose the *MFP-Tree*, a probabilistic mixed-feature indexing scheme. For the user query, there are four cases in terms of query element which are shown below. So for each image I_i, its index key can be defined as:

For case 1 (e.g., *A query image*), we have

$$key(I_i)/Sim(I_i,O_j)=vSim(I_i,O_j) \tag{8}$$
$$Conf(I_i,I_j)=1$$

For case 2 (e.g., *A query image + its style*), we have

$$key(I_i)/Sim(I_i,O_j)=w_{v_1}\times vSim(I_i,O_j)+w_{s_1}\times stSim(I_i,O_j) \tag{9}$$
$$Conf(I_i,I_j)=Conf_{ST}(I_i)\times Conf_{ST}(I_j)$$

For case 3 (e.g., *A query image + its sentiment*), we have

$$key(\lambda_i)/Sim(I_i,O_j)=w_{v_2}\times vSim(I_i,O_j)+w_{e_1}\times seSim(I_i,O_j) \tag{10}$$
$$Conf(I_i,I_j)=Conf_{SE}(I_i)\times Conf_{SE}(I_j)$$

And for the last case (e.g., *A query image + its style + its sentiment*), we have

$$key(I_i)/Sim(I_i,O_j)=w_{v3}\times vSim(I_i,O_j)+w_{s2}\times stSim(I_i,O_j)+w_{e2}\times seSim(I_i,O_j) \qquad (11)$$

$$Conf(I_i,I_j)=Conf_{ST}(I_i)\times Conf_{ST}(I_j)\times Conf_{SE}(I_i)\times Conf_{SE}(I_j)$$

Note the above weights (such as w_{v1}, w_{v2}, w_{v3}, w_{s1}, w_{s2}, w_{e1}, and w_{e2}) from Eqs. (8)-(11) can be obtained by using the multi-variable regression method introduced in Section 4.2.

Since the images are grouped into *T* clusters, to get a uniform index key of image in different clusters, the index key in Eqs. (8-11) can be rewritten by Eq. (12):

$$KEY(I_i)=j\times c+key(I_i) \qquad (12)$$

where *j* is the ID number of cluster I_i falls in and constant *c* is a large number, thus, it is guaranteed that the search range of key(I_i) is not overlapped.

To facilitate retrieving images via submitting an auxiliary information (e.g, the style, or sentiment name) of I_i with probabilistic confidence, its index keys can be rewritten in Eq. (13):

$$KEY(I_i)=j\times c+key(I_i)+Conf(I_i) \qquad (13)$$

Eqs. (8-11) show the index keys of image respectively, which correspond to four independent indexes. In order to incorporate them into an integral one, we derive a new uniform index key expression by adding four stretch constants(i.e., C_1 to C_4) shown in Eq. (14):

$$KEY(I_i)=\begin{cases} C_1+KEY(I_i) & (a) \\ C_2+KEY(I_i) & (b) \\ C_3+KEY(I_i) & (c) \\ C_4+KEY(I_i) & (d) \end{cases} \qquad (14)$$

where $C_1=0$, $C_2=1\times10^4$, $C_3=1.5\times10^4$, and $C_4=2\times10^4$. The above four constants should be set large enough to stretch the value ranges of the index keys so that they do not overlap with one another.

For an image, its two values of *StyID* and *SenID* are recorded in the corresponding index key of a MFP-Tree, which is basically a B$^+$-tree structure. Fig. 6 shows the detail steps of constructing a MFP-Tree. Note that the routine **TDis**(I_i) is a distance transformation function in Eq. (14) respectively, and **BInsert**(*key,bt*) is a B$^+$-tree insert procedure.

Algorithm 4. MFP-Tree Index Construction
Input: Ω: the image set;
Output: *bt:* the index for MFP-Tree;
1. The images in Ω are grouped into *T* clusters using the *k*-Means cluster algorithm
2. *bt*←**newFile**();   /* create index header file for MFP-Tree */
3. **for** each image $I_i\in\Omega$ **do**
4.   The distance between I_i and its centroid image are computed;
5.   Its style and sentiment are identified by user with probabilities and confidences;
6.   *KEY(I_i)=**TDis**(I_i)*;   /* Function *TDis*() is shown in Eq. (14) */
7.   **BInsert**(*KEY(I_i), bt*);   /* insert it to B+-tree */
8. **return** *bt*

Fig. 6. The index construction algorithm for MFP-Tree

4.4   Probabilistic *k*-NN Retrieval Algorithm

For *n* high-dimensional images, a probabilistic *k*-NN(PKNN) search is often a frequently used search operation, which retrieves the *k* most similar images that are closest in distance to a given image with a confidence threshold. In this section, we focus on PKNN searches of images. For example, when a user submits a query image, a threshold ε, its style name, and the sentiment name of the result images *'Vogue'* and *'Happy'*.
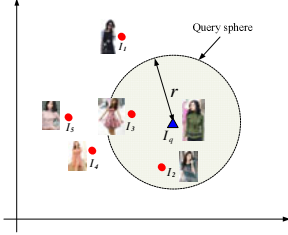

Fig. 7. The probabilistic retrieval process   **Fig. 8.** The search range in the MFP-Tree
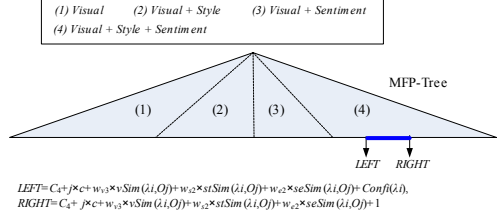
Algorithm 5. PKNN algorithm
Input: query image I_q, k, StyID or SenID, ε
Output: query results S
1. $r \leftarrow 0$, $S \leftarrow \Phi$;   //   initialization
2. **while** ($|S| < k$)   // $|S|$ refers to the number of candidate images in S
3. $r \leftarrow r + \Delta r$;
4. $S \leftarrow \textbf{\textit{RSearch}}(I_q, r, \varepsilon)$;
5.   **if** ($|S| > k$) **then**
6.   **for** $i := 1$ to $|S| - k$ **do**
7. $I_{far} \leftarrow \textbf{\textit{Farthest}}(S, I_q)$;
8. $S \leftarrow S - I_{far}$;
9. **return** S;
$RSearch(I_q, r, \varepsilon)$
10. $S1 \leftarrow \Phi$, $S2 \leftarrow \Phi$;
11. **for** each cluster sphere $\Theta(O_j, CR_j)$ and $j \in [1, T]$
12.   **if** $\Theta(O_j, CR_j)$ **contains** $\Theta(I_q, r)$ **then**
13. $S1 \leftarrow S1 \cup \textbf{\textit{Search}}(I_q, r, j)$;
14.   **end loop**
15.   **else if** $\Theta(O_j, CR_j)$ **intersects** $\Theta(I_q, r)$ **then**
16. $S1 \leftarrow S1 \cup \textbf{\textit{Search}}(I_q, r, j)$;
17. **for** each image $I_i \in S1$ **do**
18.   **if** $Sim(I_q, I_j) > r$ and $Conf > \varepsilon$ and **then** $S1 \leftarrow S1 - I_j$;   // the refinement
19. **return** S1;   // return candidate images
$Search(I_q, r, j)$
20. **if** user submits a I_q **then**
21. $LEFT \leftarrow C_1 + j \times c + vSim(I_q, O_j)$,
22. $RIGHT \leftarrow C_1 + j \times c + vSim(I_q, O_j) + 1$;
23. **else if** user submits a λ_q and its style **then**
24. $LEFT \leftarrow C_2 + j \times c + w_{v1} \times vSim(I_q, O_j) + w_{s1} \times stSim(I_q, O_j) + \varepsilon$,
25. $RIGHT \leftarrow C_2 + j \times c + w_{v1} \times vSim(I_q, O_j) + w_{s1} \times stSim(I_q, O_j) + 1$;
26. **else if** user submits a λ_q and its sentiment **then**
27. $LEFT \leftarrow C_3 + j \times c + w_{v2} \times vSim(I_q, O_j) + w_{e1} \times seSim(I_q, O_j) + \varepsilon$;
28. $RIGHT \leftarrow C_3 + j \times c + w_{v2} \times vSim(I_q, O_j) + w_{e1} \times seSim(I_q, O_j) + 1$;
29. **else if** user submits a I_q, its style and sentiment **then**
30. $LEFT \leftarrow C_4 + j \times c + w_{v3} \times vSim(I_q, O_j) + w_{s2} \times stSim(I_q, O_j) + w_{e2} \times seSim(I_q, O_j) + \varepsilon$,
31. $RIGHT \leftarrow C_4 + j \times c + w_{v3} \times vSim(I_q, O_j) + w_{s2} \times stSim(I_q, O_j) + w_{e2} \times seSim(I_q, O_j) + 1$;
32. $S4 \leftarrow \textbf{\textit{BRSearch}}[LEFT, RIGHT]$;   // the filtering step
33. **return** S2;   //   return the candidate image set

Fig. 9. The index-support PKNN algorithm

Then as shown in Fig. 8, the retrieval over the MFP-Tree in which the range is [*LEFT*, *RIGHT*], where *LEFT*= C_4+*j*×*c*+*wv3*× *vSim(I_q,Oj)*+*ws2*×*stSim(I_q,Oj)*+*we2*×*seSim(I_q,Oj)*+ε, *RIGHT*=C_4+*j*×*c*+*wv3*×*vSim(I_q,Oj)*+*ws2*×*stSim (I_q,Oj)*+*we2*×*seSim(I_q,Oj)*+1. Fig. 9 details the whole search process. Routine **RSearch()** is the main range search function which returns the candidate images of range search with centre *Iq* and radius *r* with probability larger than ε, **Search()** is the implementation of the range search. **Farthest**() returns the image which is the longest from *Iq* in *S*. **BRSearch**() is a B$^+$-tree range search function.

5 Experiments

In this section, we present an extensive performance study to evaluate the effectiveness and efficiency of our proposed retrieval and indexing method. The image data we used are garment images extracted from *Taobao.com* [12] of about 50,000 images. We have implemented an online interactive image retrieval system (see Fig. 10) to testify the effectiveness of our proposed retrieval method by comparing with a conventional one [1]. The retrieval approach and the MFP-Tree index are implemented in *C* language in which a B$^+$-tree is as the single dimensional index structure. The index page size of B$^+$-tree is set to 4096 bytes. All the experiments are run on a Pentium IV CPU at 2.0GHz with 2 Gigabytes memory. The baseline in the following experiments is a sequential scan without index support.

5.1 Effect of *m*

The experiment evaluates the effect of number of reference images(*m*) on the query precision ratio. Fig. 11 demonstrates that with the increase of the number of reference images, the precision ratio increases gradually. When the number of reference images beyond 220, the precision ratio does not increase any more. This is because the number of the affected reference images(δ) that fall in the corresponding ARIR will also increase if *m* is increased. That means, for each images, its feature probability and confidence distribution will be more accurate with the increase of δ.
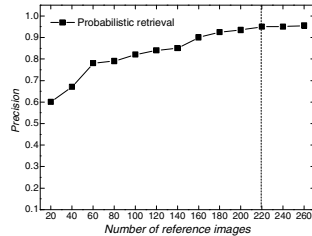

Fig. 10. One retrieval example

**Fig. 11.** Number of reference images *vs*. precision

5.2 Effect of Data Size

In this experiment, we measured the performance behavior with varying number of images. Fig. 12a shows the performance of query processing in terms of CPU cost. It

is evident that the MFP-Tree outperforms sequential scan significantly. The CPU cost of MFP-Tree increases slowly as the data size grows. It is worth mentioning that the minimal CPU cost of sequential scan can be ignored. In Fig. 12b, the experimental result reveals that the I/O cost of MFP-Tree is superior to sequential scan.
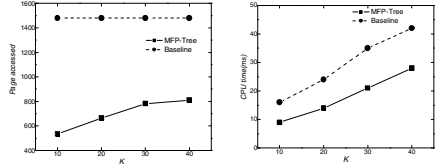
5.3   Effect of k

In the final experiment, we proceed to evaluate the effect of k on the performance of a PKNN by using the MFP-Tree. Figs. 13a and 13b both indicate that when k ranges from 10 to 40, the MFP-Tree is superior to other methods in terms of page access and the CPU cost. The results conform to our expectation that the search region of MFP-Tree is significantly reduced and the comparison between any two images is a CPU-intensive task. The CPU cost of sequential scan is ignored due to the expensive computation cost of it.
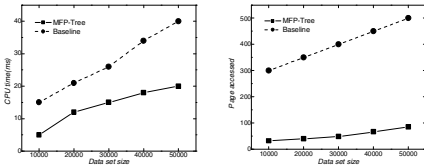
(a) CPU cost *vs.* Data size   (b) I/O cost *vs.* Data size   (a) *K vs.* Page Access   (b) *K vs.* CPU Cost

Fig. 12. Effect of data size   **Fig. 13.** Effect of *k*

6   Conclusions and Future Work

In this paper, we presented a framework to support a mixed-feature-based image retrieval process in which the probabilistic high-dimensional multi-feature indexing scheme called mix- feature-based probabilistic tree(*MFP-Tree*) is proposed to boost the retrieval performance of the large image databases. The prototype retrieval system is implemented to demonstrate the applicability and effectiveness of our new approach to image retrieval.

In our future work, we will extend our work to the social media applications such as probability-model-based personalized social media retrieval. For this kind of social retrieval, the sentimental and conceptual features of the social media objects are provided by users, which are uncertain to some degree.

References

[1] Flicker, M., Sawhney, H., Niblack, W., Ashley, J.: Query by image and video content: The QBIC system. IEEE Computer 28(9), 23–32 (1995)

[2] Virage Inc. (2005), http://www.virage.com

[3] Pentland, A., Picard, R.W., Sclarof, S.: Photobook: Content-Based manipulation of image databases. Int'l Journal of Computer Vision 18(3), 233–254 (1996)

[4] Mehrotra, S., Rui, Y., Chakrabarti, K., Ortega, M., Huang, T.S.: Multimedia analysis and retrieval system. In: Proc. of the 3rd Int'l Workshop on Multimedia Information Systems, Como (1997)

[5] Böhm, C., Berchtold, S., Keim, D.: Searching in High-dimensional Spaces: Index Structures for Improving the Performance of Multimedia Databases. ACM Computing Surveys 33(3) (2001)

[6] Guttman, A.: R-tree: A dynamic index structure for spatial searching. In: SIGMOD, pp. 47–54 (1984)

[7] Berchtold, S., Keim, D.A., Kriegel, H.P.: The X-tree: An index structure for high-dimensional data. In: VLDB, pp. 28–37 (1996)

[8] Weber, R., Schek, H., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: VLDB, pp. 194–205 (1998)

[9] Berchtold, S., Bohm, C., Kriegel, H.P., Sander, J., Jagadish, H.V.: Independent quantization: An index compression technique for high-dimensional data spaces. In: ICDE, pp. 577–588 (2000)

[10] Jagadish, H.V., Ooi, B.C., Tan, K.L., Yu, C., Zhang, R.: iDistance: An Adaptive B^+-tree Based Indexing Method for Nearest Neighbor Search. ACM Trans. on Database Systems 2(30), 364–397 (2005)

[11] Jagadish, H.V., Ooi, B.C., Shen, H.T., Tan, K.L.: Towards efficient multi-feature query processing. IEEE Trans. on Knowledge and Data Engineering 18(3), 350–362 (2006)

[12] (2010), http://www.Taobao.com

A Software Watermark Based Architecture for Cloud Security

Pengfei Dai[1], Chaokun Wang[1,2,3], Zhiwei Yu[1,4],
Yongsheng Yue[1], and Jianmin Wang[1,2,3]

[1] School of Software, Tsinghua University, Beijing 100084, China
[2] Tsinghua National Laboratory for Information Science and Technology
[3] Key Laboratory for Information System Security, Ministry of Education
[4] Department of Computer Science and Technology, Tsinghua University
{dpf09,yzw08,yys09}@mails.tsinghua.edu.cn,
{chaokun,jimwang}@tsinghua.edu.cn

Abstract. Cloud Computing has emerged as a resource sharing platform, which allows different service providers to deliver software as services. However, for many security sensitive applications such as critical data processing and e-business, we must provide necessary security protection for mitigating the threats in the shared open cloud infrastructures. In this paper, we propose a software watermark enhanced platform to assure that only the software with authorized watermark is allowed to run on the platform. Experimental results show that our architecture could provide a great protection with a small overhead.

Keywords: Cloud Computing, resource sharing platform, security protection, software watermark, Java Virtual Machine.

1 Introduction

Cloud Computing is based on the concept of service and provides shared resources, software and information to computers and other devices on-demand [1]. The cloud refers to the shared resources like the datacenter hardware and software. The benefit – economies of scale, dynamic provisioning, and low capital expenditures which could be brought by Cloud Computing now has been exemplified by Amazon's Elastic Compute Cloud (http://aws.amazon.com/ec2/) and Google App Engine (http://code.google.com/intl/en/appengine/). However, there are just two sides to every story including Cloud Computing. This sharing of resource platform along with the fact that the cloud provider may lack control over the cloud infrastructure causes tremendous security concerns about the integrity and confidentiality of a service provider's software [3]. How could cloud providers ensure that the software running on the shared platform is safe? How could cloud providers guarantee that the malicious software is not able to run in the shared platform? Many new risks need to be solved.

In this paper, we propose an architecture that could help the cloud providers to prevent the malicious software from invading. By using our watermarking

method, the software will be examined before it runs on the cloud. For example, when an attacker pretends to be a legitimate service provider to upload some malicious software to the cloud, the trusty environment we constructed will detect the software whether it has the valid watermark. One advantage of our approach is that using software watermark to ensure the security of cloud infrastructure is relatively secret and hard to be discovered. Unlike traditional user-based authentication methods, the software delivered to the cloud needs to be verified, which would make the cloud more secure.

However, it is very difficult for users to choose an appropriate watermarking algorithm to satisfy their requirements and even make good use of the algorithm, for the following reasons.

- Most of the users don't have the background knowledge of watermarking algorithm. Moreover, various algorithms have specific requirements about the software (e.g., Opaque Predicates algorithm [8] requires condition sentences).
- Generally, the implementation of a watermarking algorithm would introduce some restrictions to the usage (e.g.,SHKQ [11], which is realized in "thsstrup" (`https://sourceforge.net/projects/thsstrup`), requires the type of the watermark to be number).

In order to facilitate the users with selecting appropriate watermarking algorithms, we design a recommendation algorithm for watermarking scheme (RAWS). Compared with traditional digital signature method, our watermark scheme is relatively flexible since it could help user to select suitable algorithm according to user demand and customize the Java Virtual Machine (JVM) for different users which will be described in Section 4 and 5. Moreover, in the actual application our software watermark scheme could be used as a good supplementary of digital signature to provide better protection in the cloud. In summary, this paper makes the following contributions:

1. We design an architecture on the basis of software watermark for mitigating the risks to the cloud infrastructure.
2. We introduce RAWS for selecting the proper watermarking algorithm. Software providers just need describe their demands. Whereafter, our method automatically choose the appropriate algorithm which suits their needs.
3. We have implemented a prototype of the architecture. It can automatically embed the watermark into the software according to the users' demands, customize the JVM and then deploy the customized JVM to the cloud.
4. We have conducted both analytical study and experimental evaluation to quantify the performance of our architecture.

The rest of this paper is structured as follows. Firstly, Section 2 elaborates security risks and challenges in Cloud Computing. Section 3 redefines some fundamental terms and introduces new ones for our scheme. Then We discuss the design of our watermark based architecture in Section 4. A detailed introduction about RAWS is given in Section 5. Section 6 discusses the prototype implementation of our architecture. We provides the analytical and experimental evaluation results in Section 7. Section 8 surveys the related work. Finally, we conclude our contributions and discuss the future work in Section 9.

2   Threat Model in Cloud Service

In Cloud Computing, software from different service providers but sharing the same platform is supported by the Platform as a Service (PaaS) model provided by cloud systems. However, the shared infrastructure inevitably introduces new security risks. We analyze these security risks according to the perspectives of various roles that are involved in the sharing activity.
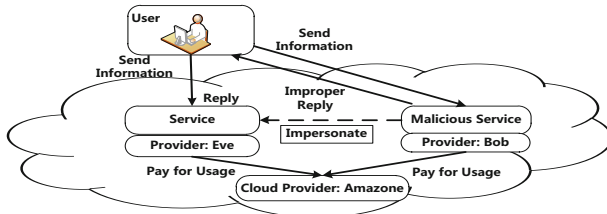

Fig. 1. Security Threat to Cloud Service

Cloud Provider.   The cloud provider shoulders the risks that the cloud infrastructure would be misused by the malicious software. External malicious software may invade into the cloud and also attackers may pretend as the service providers to corrupt the cloud from inside. As shown in Figure 1, Bob who is the attacker deploys his software to the cloud. Hence Bob has the possibility to violate the cloud. Even more, Bob could monitor the activities in the cloud, utilize side channels and covert channels for reverse engineering, infiltration, exfiltration, certain kinds of encryption cracking, and other attacks [2].

Service Provider.   When a service provider deploys her/his software to the cloud, the only expense is on leasing the resource of cloud provider. But if a malicious software pretends as the legitimate software running on the same hardware platform with the service provider's software, it would get the chance to invade into the the service provider's virtual environment [4]. Take the worm virus as a example. If it invades into the service provider's virtual environment and runs with the service provider, it could generate many copies which occupy so abundant resources that the service could not run properly.

Service User. Since the service on the cloud shares to the public, the sensitive information would increase amazingly with the growth of the service users in that the users have to submit their information to the service for applying it. Take paypal (`https://www.paypal.com/`) for instance, we need to submit our accounts information to apply the service. As showed in Figure 1, Alice, who applies the service provided by Eve in a cloud, submits her sensitive information to the service. The information would be transferred to the cloud. However, if Bob's service impersonates Eve's service, the senfdskajflkjadsjlksitive message may leak to Bob or improper reply from Bob's service may transfer to Alice.

3   Basic Concepts

A formalization of software watermark has been proposed by [5]. However, in order to give a precise description of our watermarking scheme, we redefine some fundamental terms and introduce new ones as follows.

Definition 1 (Configuration Profile). *A configuration profile is a message which contains the information including watermarking algorithm, watermark, key and software provider's information. We define the set of configuration profiles as **P**. Profile is short for configuration profile in this paper for convenience.*

Table 1 illustrates an example of configuration profile. Opaque Predicates is the watermarking algorithm we choose for the software provider. Watermark and key are 652272 and 1234, respectively. We fill Provider's Info with the software provider's information and here the information is ABC_Company.

Table 1. An Example of Configuration Profile

Algorithm	Watermark	Key	Provider's Info
Opaque Predicates	652272	1234	ABC_Company

Definition 2 (Profile Generating). *Let **T** denote the set of TrustInfo which includes the software provider's authentication information and security demands, **S** the set of software submitted by the software provider. We call a function $Ge : \boldsymbol{S} \times \boldsymbol{T} \to \boldsymbol{P}$ a profile generator. If $P = Ge(S, T)$ for $S \in \boldsymbol{S}, T \in \boldsymbol{T}$, then P is a profile for embedding and checking watermark of S.*

Definition 3 (JVM Customizing). *We call a function $Cu : \boldsymbol{JVM} \times \boldsymbol{P} \to \boldsymbol{JVM}$ a customizer. If $JVMp = Cu(JVM, P)$ for $JVM \in \boldsymbol{JVM}$ and $P \in \boldsymbol{P}$, then JVMp gets the ability to check the watermark in the software corresponding to the profile P, we call JVMp the customized JVM corresponding with P.*

4   The Architecture

In this Section, we describe the design and implementation of our architecture. First, we give an overview of our approach in Section 4.1. Then two parts of the architecture – software trust manage center (STMC) and software watermark embedding server (SWES) will be discussed in Section 4.2 and 4.3, respectively.

4.1   Overview of the Architecture

As shown in Figure 2, software providers firstly upload their software and TrustInfo to STMC. Then the profile is generated by the STMC according to the software and TrustInfo. Subsequently, the STMC customizes the JVM depending on the profile and deploys JVMp to the cloud. Meanwhile, the STMC transfers
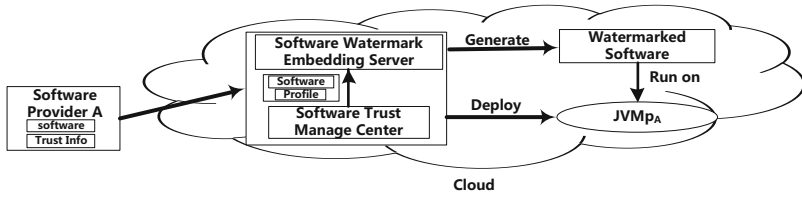
Fig. 2. System Overview

the software and profile to STMC for embedding watermark to the software. Thus, the watermarked software can run on the cloud after being detected by the JVMp and the malicious one can be kept away from the cloud. The $JVMp_A$ is deployed only for the software provide A in Figure 2.

4.2   Software Trust Manage Center

The STMC is the kernel of our architecture. Firstly, the software provider needs to connect with the STMC using her/his ID and password. Then the STMC matches the TrustInfo (defined in Section 3) of the provider to verify the "Request". Here the verification information is stored in the Software Provider Info LIB. If authentication failure occurres, the software provider's request will be rejected. When the authentication for the request is successful, the Configuration Profile Manage Center will get the software and TrustInfo as input and employ RAWS to generate the profile. According to the generated profile, Java Virtual Machine Configurator (JVMC) firstly gets an extracting module from the Watermark Extracting Module Generator. Subsequently, the JVMC invokes some "Auxiliary Programs" (shown in Figure 3) to customize the original JVM. In addition, special process policy, which are used to tell the JVM the rule of dealing with the software with invalid watermark, is appended to the JVM. Finally, after the JVM has been customized, we deploy them to the cloud.
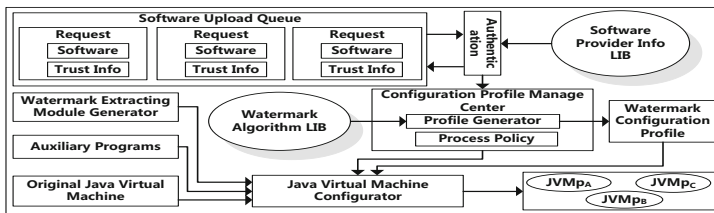

Fig. 3. Software Trust Manage Center

4.3   Software Watermark Embedding Server

As shown in Figure 4, SWES maintains a plug-in LIB in which a number of software watermarking algorithms are realized as plug-ins. When the software
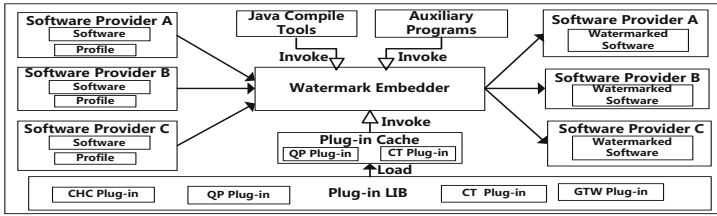
Fig. 4. Software Watermark Embedding Server

comes with a profile, the watermark embedder loads the algorithm plug-in from the plug-in LIB based on the profile to the plug-in cache. The watermark embedder subsequently invokes the Java compile tools, some auxiliary programs and algorithm plug-in to embed watermark into the software.

5   Recommendation Algorithm for Watermarking Scheme

In this Section, we give a detailed explanation about RAWS. A formal specification is given for RAWS as follows:

Let $S_i(m \geq i \geq 1)$ denote the attribute set and $T_i(n \geq i \geq 1)$ denote the target set, and then we get the mapping:

$$S_1 \times S_2 \times, ..., \times S_m \to \langle T_1, T_2, ..., T_n \rangle$$

If $s_i \in S_i$ for $m \geq i \geq 1$ and $t_j \in T_j$ for $n \geq j \geq 1$ then for $n \geq k \geq 1$, f_k is a function: $t_1 = f_1(s_1, s_2, ..., s_m), t_2 = f_2(s_1, s_2, ..., s_m), ..., t_n = f_n(s_1, s_2, ..., s_m)$.

Table 2. Watermarking Algorithm Table

Algorithm	Security	Efficiency	Constraint Condition
Add Instruction	0	2	Watermark and Key:$< -128, 127 >$
Add Local Variable	0	2	
Add Method	0	2	Watermark and Key:$< Strings >$
Add Field	0	2	Watermark and Key:$< Strings >$
Add Class Attribute	0	2	Watermark and Key:$< Strings >$
Register Allocation	1	2	Watermark and Key:$< Numbers >$
Spread Spectrum	1	2	
Opaque Predicates	1	1	Watermark and Key:$< Numbers >$
Semi-Dynamic Multiple	2	1	Need Extra Parameters m,n

Note: 0,1,2 means different levels of security or efficiency.

Based on the formal specification, we discuss the concrete implementation of RAWS. The target set in our implementation is the customer satisfaction degree denoted as T_1. There are two attribute sets: "security" attribute set and "efficiency" attribute set denoted as S_1 and S_2, respectively. Table 2 contains the values of "security" attribute set and "efficiency" attribute set for different watermarking algorithms. For the dynamic software watermarking algorithm [13], the watermark can only be expressed when the software is running. However,

---

Algorithm 1. $Ge(s, t)$

Input: s - s for software uploaded by the provider; t - t for TrustInfo of the provider;
Output: the profile;
 1: A = all the watermarking algorithms supported by our architecture
    {compute the W_s and W_e according to the demand from TrustInfo of provider}
 2: $CalWeightByDemand(t.demand, W_s, W_e)$ {$t.demand$: the demand of the software provider}
    {eliminate the algorithms according to software characteristic}
 3: **if** !(s contains condition sentences) **then**
 4:    Eliminate opaque predicates algorithm from A
 5: **end if**
 6: **if** !(s contains program cut points) **then**
 7:    Eliminate semi-dynamic multiple algorithm from A
 8: **end if**
    {compute t_1 of each candidate algorithms and get the most proper algorithm $ProperA$ from A}
 9: $MaxT = 0$
10: $ProperA = NULL$
    {s_1^a, s_2^a are the value of $Security$ set and $Efficient$ set corresponding to a}
11: **for** each $a \in A$ **do**
12:    $t_1 = W_s \times s_1^a + W_e \times s_2^a$
13:    **if** $MaxT < t_1$ **then**
14:        $MaxT = t_1$
15:        $ProperA = a$
16:    **end if**
17: **end for**
18: $constraints = getConstraint(ProperA)$ {get constraint condition of $ProperA$}
19: $profile = generateProfile(ProperA, constraints)${generate configuration profile}

---

it will increase the risks to the cloud if the software is allowed to run before being checked. Thus, we do not bring dynamic watermarking algorithms in the table. The watermarking algorithms in our architecture come from "thsstrup". As shown in Algorithm 1 the function f_1 is: $t_1 = W_s \times s_1 + W_e \times s_2$. W_s and W_e are the weights for the "security" set and the "efficiency" set, respectively. Since it's hard to fix the actual values of W_s and W_e, a number of experiments have been done to get the optimal values of them. When a user gives her/his demands, the suitable algorithm is chosen according to the value of t_1.

In addition, the statistic in Table 2 is estimated according to the relevant articles [9,13].Since different watermarking algorithms have different constraint conditions in their implementations (e.g., the key and watermark of "Add Instruction" should be numbers, even more the numbers should be in the range of [-128, 127]), we arrange these constraints in the watermarking algorithm table.

6   Prototype Implementation

As shown in Figure 5, we have implemented a prototype of our architecture. While the current implementation works on the Ubuntu 9.04, its core component is OS-independent and can be easily ported to other operating systems. In the following, we briefly describe some non-trivial implementation details.

First, we realize several software watermark algorithms. A variety of software watermark algorithms are proposed in the literature, but related implementation is few. In our architecture, more than ten algorithms have been realized.

Second, based on the design of recommendation algorithm, we realize the RAWS for choosing watermark algorithm in our architecture.
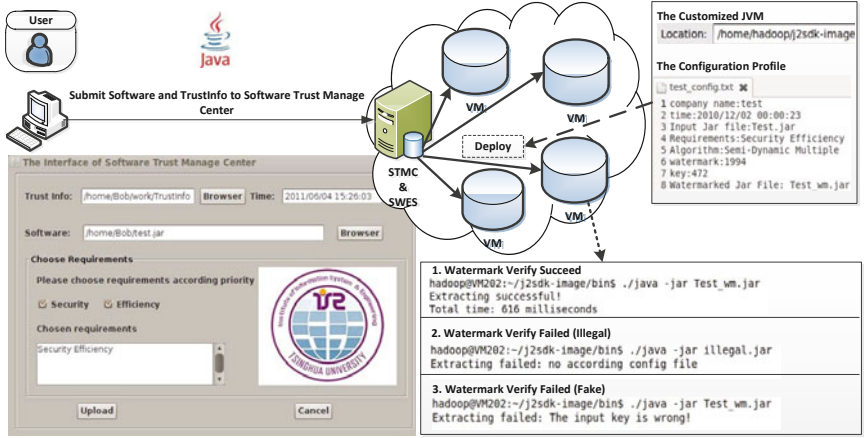
Note: Illegal: the verified software has no related information in the configuration profile; Fake: the verified
software has the same name as the valid software

Fig. 5. Prototype of Our Architecture

Third, the implementation of the customized JVM is based on default JVM in OpenJDK 1.7.0. Various hooks are added to the default JVM. For instance, a hook is added so that, when one software needs to run on the customized JVM, the JVM transfers the control to the watermark extracting module by loading a native library. The module is compiled separately from the JVM into a shared library which the JVM loads when corresponding events trigger. Our design tries to minimize the changes to the OpenJDK, and thus has the advantage of reducing the development costs since OpenJDKs re-compilation delay is significant.

7   Experimental Evaluation

In this Section, we evaluate three key aspects to demonstrate the performance of our system. Firstly, we evaluate RAWS's performance by comparing with random selection of watermarking algorithms. Secondly, based on various sizes of Java programs, we compare the time cost for embedding and checking the watermark, respectively. Thirdly, we measure the performance of the program "wordcount" with watermark on the watermarked hadoop (http://hadoop.apache.org/).

7.1   RAWS Performance

We define four types of demands such as "Security", "Efficiency", "Security (First), Efficiency (Second)" and "Efficiency (First), Security (Second)" to investigate the effectiveness of RAWS. Based on the design and implementation of the algorithms in "thsstrup", we know that "Semi-Dynamic Multiple" is the safest watermarking algorithm among the algorithms in Table 2 and also the
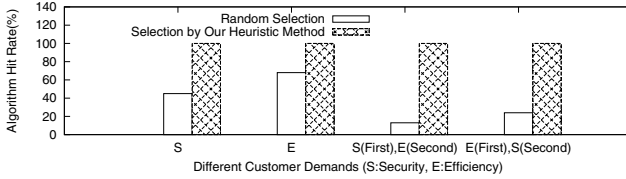
Fig. 6. Algorithm Hit Rate between Random Selection and RAWS

most appropriate algorithm to meet the demand: "Security (First), Efficiency (Second)". For demand: "Efficiency", "Add Instruction" and "Add Local Variable" are almost the same. For the sake of contrast, we choose the algorithms for 100 Jar programs in two ways: Random Selection and RAWS. If the suitable algorithm has been selected, we score one point. Otherwise, the score is zero. After the algorithms for the 100 Jar programs have been chosen, we sum the scores and compute the algorithm hit rate: $AlgorithmHitRate = sum(score) \div 100 \times 100\%$. As shown in Figure 6, RAWS has a better performance comparing to the random selection. Especially, when the demand is "Security (First), Efficiency (Second)", the algorithm hit rate for random selection is lower than 20%.

7.2   Watermark Time Cost

In this experiment, we run a machine with a Pentium(R) dual-core E5300 2.60 GHz CPU and 2GB RAM. The operation system is Ubuntu 9.10 with 2.6.31-22-generic kernel. Opaque predicates algorithm is used to embed watermark. As shown in Figure 7, the time cost for embedding watermark increases with the growing of the Java software's size. For a software with the size of 1000KB, the embedding time is less than 5 seconds. So, we could know that the time cost for embedding watermark is in an acceptable range. In order to evaluate the cost on verifying the software, a test is set up on Java Software with the watermark embedded in the prior experiment. As shown in Figure 8, the checking time is relatively short. For a 2 MB Jar file, the time cost for checking is just 1.5 seconds. So, the time cost for checking the watermark is relatively small and acceptable.
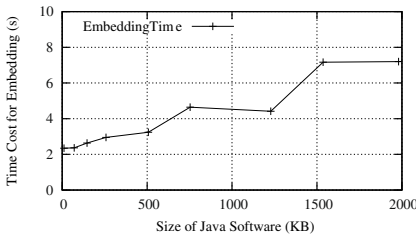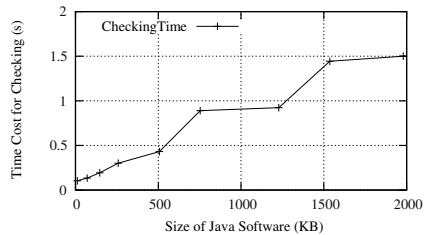

Fig. 7. Time Cost for Embedding

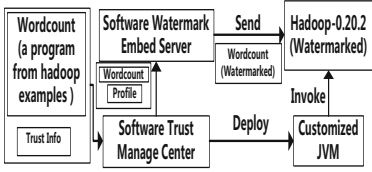**Fig. 8.** Time Cost for Checking
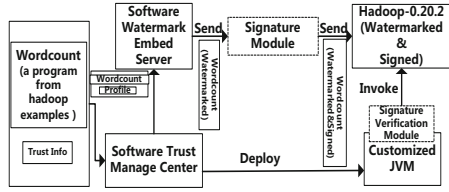
Fig. 9. Process for Wordcount

Fig. 10. Process for Wordcount with Signature

7.3   Watermark Application Performance

The experiment is based on hadoop. We have five nodes in the hadoop cluster, with each node processor Pentium (R) Dual-Core CPU E6300 2.8GHz, 4G memory, 800G hard disk. The nodes provide services as slavers, one within which deployed as the master. Operating systems are Ubuntu 9.10. The sample software is the "wordcount" from the hadoop examples. As shown in Figure 9, the "wordcount" with TrustInfo firstly submitted to the STMC for generating the profile. The STMC subsequently automatically customizes the JVM according to the generated profile. Meanwhile, the SWES loads "wordcount" with the profile to construct the "wordcount" with watermark. Finally, we evaluate the performance of watermarked "wordcount" by running it on the watermarked hadoop.

In order to illustrate that our watermark scheme doesn't conflict with signature method. A simple implementation of combining signature method and our scheme is realized in Figure 10. Before the watermarked "wordcount" is sent to Hadoop, a signature module is added to sign the software. And our customized JVM integrated with a signature verification module can verify the signature and watermark step by step to present illegal software running on the cloud.
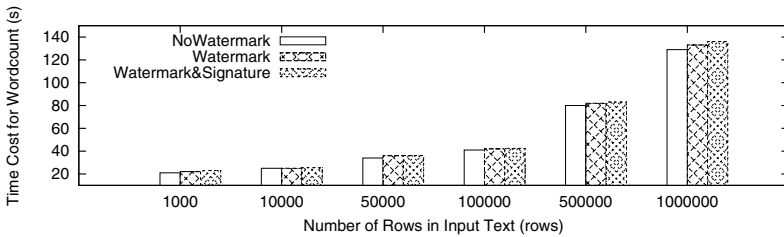

Fig. 11. Time Cost for Wordcount in Hadoop

According to different rows of the input text, we compare the performance of the original "wordcount", and the one with watermark running on different JVM. As shown in Figure 11, the maximum gap between two scenario appears in the rows number of 1,000,000, and the gap is just 4 seconds. When the row number of input rows is 10,000, the time cost of the "wordcount" running on the normal hadoop is equal to the "wordcount" with watermark in our architecture. We speculate the experimental data has fluctuation due to the problem of I/O

interference between virtual machines [1]. The experiment results of the "word-count" with watermark and signature indicate that our watermark scheme and traditional signature method could be combined to protect the cloud together and the cost which is brought by our architecture is relatively low.

8   Related Work

In this Section, we survey the related work in detail from two aspects. One aspect is about current developing situation about Cloud Computing Security. The other is about the research situation of software watermark in cloud and the comparison between software watermark and digital signature.

While the adoption of Cloud Computing in the economic field becomes more widespread, the security challenges it poses becomes more striking. Thus, various aspects of researches about Cloud Computing security have been identified and studied, including data outsourcing and data privacy [10], access control [7].

There are already some studies about software watermark in the cloud. Yu et al. apply software watermark to MapReduce to keep the software's safe [12]. Fu et al. propose software watermark to ensure the security of the cloud platform [6]. Digital signature as an authentication technique is also applied to the JVM. Compared with signature, our scheme simplifies the process of selecting algorithm (detailed discussing in Section 4). Meanwhile, since different JVM is customized for every user, it ensures that even if an attacker sneaks into one valid user's JVM, (s)he is still unable to hack into other users'. Furthermore, in the actual application signature and our watermark scheme don't conflict with each other, and could be combined to ensure the security of the cloud together.

9   Conclusion

While Cloud Computing which emerges as a resource sharing platform brings many benefits, it also introduces significant security concerns. In this paper, we analyze these security risks according to the perspectives of various roles and propose an architecture which employs software watermarking methods to reduce the threats. In this architecture, RAWS is designed to choose the appropriate watermarking algorithms based on the user's requirements and the JVM is customized to verify the software's validity in the cloud. Our experimental results show that the architecture is feasible and imposes low performance slowdown.

So far, our proposal and implementation are just for protecting Java programs in the cloud. However, in our further research, we would extend software watermark to protect more software applications such as C/C++, Python etc.

References

1. Armbrust, M., et al.: Above the Clouds: A Berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley (February 2009)
2. Blumenthal, M.S.: Hide and seek in the cloud. IEEE Security & Privacy 8(2), 57–58 (2010)
3. Chen, Y., Paxson, V., Katz, R.: What's new about cloud computing security. University of California, Berkeley Report No. UCB/EECS-2010-5 January 20(2010), 2010–5 (2010)
4. Cloud Security Alliance (March 2010)
5. Collberg, C., Thomborson, C.: Software watermarking: models and dynamic embeddings. In: POPL 1999: Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp. 311–324. ACM, New York (1999)
6. Fu, J., Wang, C., Yu, Z., Wang, J., Sun, J.: A watermark-aware trusted running environment for software clouds. In: 2010 Fifth Annual ChinaGrid Conference (ChinaGrid), pp. 144–151. IEEE (2010)
7. Hu, L., Ying, S., Jia, X., Zhao, K.: Towards an Approach of Semantic Access Control for Cloud Computing. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) Cloud Computing. LNCS, vol. 5931, pp. 145–156. Springer, Heidelberg (2009)
8. Monden, A., Iida, H., Ichi Matsumoto, K., Torii, K., Inoue, K.: A practical method for watermarking java programs. In: COMPSAC, pp. 191–197 (2000)
9. Nagra, J., Thomborson, C.D., Collberg, C.S.: A functional taxonomy for software watermarking. In: ACSC, pp. 177–186 (2002)
10. Pearson, S., Shen, Y., Mowbray, M.: A Privacy Manager for Cloud Computing. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) Cloud Computing. LNCS, vol. 5931, pp. 90–106. Springer, Heidelberg (2009)
11. Stern, J.P., Hachez, G., Koeune, F., Quisquater, J.-J.: Robust Object Watermarking: Application to Code. In: Pfitzmann, A. (ed.) IH 1999. LNCS, vol. 1768, pp. 368–378. Springer, Heidelberg (2000)
12. Yu, Z., Wang, C., Thomborson, C., Wang, J., Lian, S., Vasilakos, A.: A novel watermarking method for software protection in the cloud. Software: Practice and Experience (2011) (accepted)
13. Zhu, W., Thomborson, C., Wang, F.-Y.: A Survey of Software Watermarking. In: Kantor, P., Muresan, G., Roberts, F., Zeng, D.D., Wang, F.-Y., Chen, H., Merkle, R.C. (eds.) ISI 2005. LNCS, vol. 3495, pp. 454–458. Springer, Heidelberg (2005)

FindCredPg: A Novel Method to Find Credible Pages Based on Trust Web Graph

Teng Wang[1,2], Qing Zhu[1,2], Shan Wang[1,2], and JingFan Liang[1,2]

[1] Key Laboratory of the Ministry of Education for Data Engineering and Knowledge Engineering, Renmin University of China, Beijing, China
[2] School of Information, Renmin University of China, Beijing, China
{wangteng,qz,swang,liangjinfan}@ruc.edu.cn

Abstract. Finding credible pages is a challenging problem on the Web. Our key observation in this paper is that *credible pages usually link to credible content-related pages*, which is different from *a normal page usually links to normal pages* in spam page detection. We propose a novel method to find credible pages based on the trust web graph we define. This method first measures the content correlation between pages connected by hyperlinks, then it combines web link structure with content correlation value of pages to build a trust web graph. At last, credible pages are found successfully by using trust relation of vertices on the trust web graph. We construct a real-world data set by crawling millions of pages on the web and run a set of experiments on this data set. Experiment results show that the accuracy of this method is near 80% and the efficiency is higher.

Keywords: Credibility, Hyperlink, Content correlation, Trust web graph.

1 Introduction

People suffer from non-credible pages. Browsing pages is the most common way for people to get information. By using search engines, people can find pages which they want to browse. Unfortunately, the searching results are not always credible. People usually click a hyperlink on a page, and then access to another page. In many cases, hyperlinks don't point to a credible pages. Non-credible pages mislead web users and pose serious impacts on user experience or their real-world life. For instance, since malicious sites, such as phishing sites, fetch people's account information, billions of dollars annually is lost [1].

How to judge whether a page is credible or not is a critical problem. Some researchers use link structure of web to compute a page's credibility scores and then assess the page[2] [3]. The basic idea of these works is *normal pages are likely to link to normal pages*. But in some cases, these methods do not work well. For example, *www.39.net* is a reputable health site in China. But along the hyperlinks belonging to *www.39.net*, some non-credible pages are reached, such as *www.jingduho.cn*. Our key observation is that most hyperlinks created by credible pages help users reach credible content-related pages. In addition, these works are based on pagerank. The number of hyperlinks belonging to a page is an indispensable factor during computing credibility scores. For example, given a page p which contains n hyperlinks pointing to page p_1, p_2,..., p_n

respectively. If p is credible, from p's perspective, the credibility probability of $p_i(1 \leq i \leq n)$ is $\frac{1}{n}$, which heavily depends on the number of hyperlinks contained by p. But in reality, little interaction between the hyperlinks belonging to the same page is found. So whether p_i is credible or not is unrelated to other hyperlinks belonging to p, excluding the one pointing to p_i. Moreover, these works define non-credible pages in terms of web spam, not in terms of information quality.

In this paper, we focus on page-level credibility and propose a novel method to compute pages' credibility scores based on the trust web graph we define. In our method, we define credible pages in terms of information quality, rather than in terms of web spam. And the hyperlinks contained by the same page are considered independent. Through analyzing the relations between page keywords and page title, we propose a method based on page keyword and page title to measure the content correlation value between two pages connected by a hyperlink. Usually, $p \rightarrow q$ means p agrees or trusts q to some extent. The content correlation of p and q is used to measure p's confidence in q, which is also called trust value in the trust web graph we define. We construct the trust web graph using pages as vertices, hyperlinks as edges and trust values as the weights of corresponding edges. Then, trust relation of vertices on the trust web graph is used to compute credibility scores of a page. At last, we can judge whether the page is credible or uncertain based on its credibility scores. In our experiments, a small seed set of credible pages should be chosen firstly, since the seed set is the start point of this method. In addition, some credible pages and uncertain pages are picked out as test data set. Then, we evaluate this method in terms of accuracy and efficiency. The results show the accuracy of this method is near 80% and the efficiency of the method is higher.

In summary, the contributions of this paper are:

- We propose a novel method to find credible pages through computing pages' credibility scores based on the trust web graph we define.
- We construct a trust web graph, then implement our proposal through using trust relation of vertices on the trust web graph.
- We construct a real-world data set by crawling millions of web pages, and a set of experiments are run on the data set. Experiment results show the advantages of our method.

The rest of this paper is organized as follows. Section 2 summeriness related works. Section 3 explains preliminaries of this studying. In Section 4, we show our proposed solution. Section 5 shows the experiment results and the analysis. At last, section 6 concludes this paper.

2 Related Works

In [4], the authors define credibility as believability. They identify two key components of credibility: trustworthiness and expertise. Trust worthiness is defined by the terms such as well-intentioned, truthful, unbiased, and so on. And expertise is defined by the terms such as knowledgeable, experienced, competent, and so on. In [5], authors give out three categories features: on-page, off-page, and aggregate features, and explore how these features influence assessing pages. The above works explore how people

assess a page and what are considered by people during assessing a page. These works emphasize on the process of page assessing.

Some researchers believe if a page is non-credible, some style-related features of the page will appear. In [6], the authors provide some style-related features to assess a page in Wikipedia. A similar idea is used in spam page detection [11]. In addition, several researchers develop algorithms to predict the credibility of a page. TrustRank [2] and CredibleRank [3] are better known algorithms. In TrustRank and CredibleRank, the link structure of web is used to determine credibility scores of pages. In [2], authors propose trust rank based on "normal pages usually link to normal pages". First, several reputable pages are selected as seeds, and the credibility scores of a page in the seeds is set to 1. Then trust dampening and trust splitting are used to compute the credibility scores of other pages. According to credibility scores of pages, normal pages are selected. In [3],the concept link credibility is introduced. The authors separate link credibility from page quality. By credibility-based link analysis, the normal pages are picked out. The two algorithms define non-credible pages in terms of web spam, not in terms of information quality.

Recent years, some researchers focus on the credibility in information item level, rather than page-level. In [7][8], how to assess fact statements is discussed. In [9], a method to rate the credibility of news articles is proposed. In [10], authors propose a model to excavate real-world events from the Web and offer users reliable evidence of an event. These works' target is to find credible pages in terms of information quality, but they have not considered the link structure of web.

3   Preliminary

3.1   Notation

In this paper, hyperlink h is a link from a page p pointing to a page q. Basically, hyperlinks are in two categories: site inside link and inter site link. A link which connects pages belonging to the same site is called site inside link. Inter site link refers to the link which connects pages from different sites. Since if a site is credible, the pages of the site are considered credible. We focus on inter site links. Credibility-status of a page is used to describe whether the page is credible or not in this paper. Pages are divided into two categories based on corresponding credibility-status: credible page and uncertain page. We refer these pages, which can not be proved credible by enough evidence, as uncertain pages. In this paper, We use S_C and S_N to denote credible page set and uncertain page set, respectively.

We use $P_p \leq 1$ denote the credibility scores of page p. Given a threshold δ, if $P_p \geqslant \delta$, p is regarded as a credible page. If $P_p \leqslant \delta$, p is regarded as an uncertain page. Credibility scores of a page and δ decide the credibility-status of the page.

3.2   Trust Web Graph Model

A trust web graph $TG = (V, E, T)$ is a weighted directed graph. V is a set of vertices, E is a set of edges and T is a set of trust values between corresponding vertices.

$V = v_1, v_2, \ldots, v_n$ and $v_i (1 \leqslant i \leqslant n)$ represents page p_i. If $p_i \to p_j$, there is an edge from v_i to v_j, which is denoted by $e_{i,j} (e_{i,j} \in E)$. The trust value between v_i and v_j describes v_i's confidence in v_j. We use the trust values between vertices as the weights of corresponding edges. $t_{i,j}$ is used to denote the trust value of v_i and v_j. If $e_{i,j} \in E$, $t_{i,j}$ is equal to the content correlation value of p_i and p_j. Otherwise, $t_{i,j} = 0$.
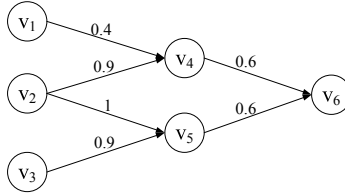

Fig. 1. A trust web graph

Fig.1 shows an example of trust web graph. Here, v_i represents page $p_i (1 \leqslant i \leqslant 6)$. $e_{1,4}$ means $p_1 \to p_4$. The weight of $e_{1,4}$ is 0.4, which means $t_{1,4} = 0.4$.

4   Proposed Solutions

The goal of this paper is to find credible pages, rather than non-credible pages. The procedure of FindCredPg is follow: firstly, since the trust relation between vertices on the trust web graph is based on the content correlation of corresponding pages, we need compute the content correlation value of corresponding pages; secondly, we construct a trust web graph; at last, we use trust relation of vertices to get the credibility scores and credibility status of the page needed to be determined.

4.1   Computation of Content Correlation Value

In this paper, the content correlation of pages describes their relevance in terms of topics, rather than the similarity of their meanings. For example, p and q are related to hospital A. p's contents are positive. q's contents are negative. Since their topics are the same, the relevance of p and q is high. Even though their meanings are totally different. Here, $R_{p,q}$ denotes the content correlation of p and q. Especially, $R_{p,q}$ is one-way, it means $R_{p,q} \neq R_{q,p}$. Given p involves topic t_1 and t_2, while q involves topic t_1, t_2 and t_3. It is obvious that p involves a part of q. On the contrary, q involves the whole p. So, from p's perspective, $R_{p,q} = 1$. But from q's perspective, $R_{q,p} \lneq 1$.

Since analysis of page contents is difficult, we propose a method to compute the content correlation of corresponding pages based on page keywords or title. This idea is reasonable for the following reasons. Page keywords are important for a page, they reveal the contents of the corresponding page. Moreover, page keywords are the main basis for classification of pages. Usually, page keywords is a collection of words and phrases and they are always arranged carefully to reveal the page contents, in order

to be easily searched by search engines. That is to say, we can grasp page contents by getting corresponding page keywords. Furthermore, some characteristics of page keywords make the computation easily and efficiently, since keywords almost don't contain stop words, exclamation mark, question mark, etc; and pages about the same topics usually contain the same keywords. Besides page keywords, page title is another important description of page content. Compared to page keywords, page title is usually a statement, rather than a collection of words or phrases. In fact, not every page contains keywords. So, for these pages whose titles are missed, the titles are considered during computing corresponding content correlation values.

In Chinese, a phrase can be broken down into multiple words. If two phrases contains the same words, they are usually related. For example, given two Chinese phrases *KouQiangZhenSuo*, and *KouQiangYiSheng*, which mean *dental clinic* and *dentist* respectively, the relevance between the two phrases is always manifested by their common parts. In this example, the common part is *KouQiang*, which means *oral*. From the common part, we know the two phrases are related. So two phrases, which are not equal, may be related. In order to find the common part between phrases and get more accurate correlation value, phrase segmentation is needed. In this studying ,we use IK Analyzer[1] to realize phrase segmentation.

We compute content correlation of pages as follows. First, corresponding page keywords and page title are extracted. If a page has not contained page keywords, the title of the page is used instead of page keywords. Since, in our view, page keywords reveal page content better. Then, IK Analyzer is used to process page keywords or page title. Here, p_K and q_K represent the keywords of p and q respectively. We use p_T and q_T to denote the titles, whose emotional words, stop words are deleted, of p and q. After phrase segmentation by Ik Analyzer, $p_K = \{p_{k1}, p_{k2}, \ldots, p_{kn}\}$, $q_K = \{q_{k1}, q_{k2}, \ldots, q_{km}\}$. Similarly, $p_T = \{p_{t1}, p_{t2}, \ldots, p_{tn}\}$ and $q_T = \{q_{t1}, q_{t2}, \ldots, q_{tm}\}$. $R_{p,q}$ can be computed by equation 1.

$$R_{p,q} = \frac{\mid p_{K/T} \cap q_{K/T} \mid}{\mid p_{K/T} \mid} \tag{1}$$

In equation 1, $R_{p,q}$ is one-way, that is to say, $R_{p,q} \neq R_{q,p}$. If $p_{K/T} \cap q_{K/T} = \Phi$, then $R_{p,q} = 0$. Especially, if $q_{K/T} = \Phi \vee p_{K/T} = \Phi$, $R_{p,q} = 0 \wedge R_{q,p} = 0$.

4.2   Construction of Trust Web Graph

We construct the trust web graph as follows. First, we set the trust web graph to be empty. Second, we choose some pages as seed set and add corresponding vertices on the trust web graph. Third, the pages which are linked by seed pages are put in seed set, and corresponding vertices are added on the trust graph. At the same time, corresponding edges are added on the trust web graph. Forth, we repeat the third step until the graph reaches requirements. At last, we compute weights of edges of the graph.

---

[1] http://code.google.com/p/ik-analyzer/

4.3 Finding Credible Pages

In the above subsections, we give out how to construct the trust web graph. In this part, we describe how to find credible pages based on the trust web graph.

Naive Method. Credibility scores of a page are determined by the pages linking to it. Initially, a set of credible pages belonging to reputable sites are selected and put in S_C as seeds. Other pages are put in S_U. The credibility scores of pages in S_C are set to 1, while the credibility scores of pages in S_U are unknown. The seeds are the start point of our algorithm. Given a page p_u, $p_u \in S_U$. The credibility scores of p_u is determined by the pages which link to p_u. By computing the contributions of these pages to p_u's credibility scores, we can get P_{p_u} and judge whether the p_u is credible or uncertain. On the trust web graph, the vertex representing a seed page is called as seed vertex. Here, if vertex v_i represents page p_i, we define $P_{v_i} = P_{p_i}$.

We use C_{v_i,v_j} to denote the contribution of v_i to the credibility scores of v_j. If $e_{i,j}$ is exists, we define $C_{v_i,v_j} = P_{v_i} * t_{i,j}$. Given two nodes v_1 and v_3, which represent page p_1 and p_3. If $p_1 \rightarrow p_3$ and $t_{1,3}$ is known. According to trust relation between p_1 and p_3, $C_{v_1,v_3} = P_{v_1} * t_{1,3}$. If there are another node v_2 and an edge $e_{2,3}$, $C_{v_2,v_3} = P_{v_2} * t_{2,3}$. If v_3 only have edges with v_1 and v_2, $P_{v_3} = C_{v_1,v_2} + C_{v_2,v_3}$.

If q is linked by $p_1, p_2, \ldots, p_n$. q is represented by v_o and v_i represents p_i. we define an equation to compute P_{v_o} as below:

$$
P_{v_o} =
\begin{cases}
\sum\limits_{i=1}^{n} P_{v_i} * t_{i,o} & \text{if } \sum\limits_{i=1}^{v} P_{v_i} * t_{i,o} \lesssim 1 \\[2ex]
1 & \text{if } \sum\limits_{i=1}^{v} P_{v_i} * t_{i,o} \geq 1
\end{cases}
\tag{2}
$$

Since $P_{v_o} = P_q$, so according to equation 2, P_q is gotten. Given a threshold δ, if P_q is more than or equal to δ, p is considered credible. While, if P_q is less than δ, p is considered as uncertain, since we haven't enough evidence to prove it non-credible.

Algorithm 1 shows the procedure of assessing a page. The input of the algorithm is the trust web graph $TG(V, E, T)$, δ, objective page p_o and v_o representing p_o, while the output is the credibility-status of p_o. First, we set $P_{v_i} = 1$ and put v_i into a set S, if v_i is a seed vertex. On the contrary, if v_i is not a seed vertex, we set $P_{v_i} = 0$. Second, if $v_o \in S$, according to P_{v_o} and δ, the credibility-status of p_o is gotten and the algorithm is over. If $v_o \notin S$, we find the vertices which are pointed by the vertices in S, compute the credibility scores of these vertices and put them in S. Third, the second step is repeated, until the algorithm is over.

Fig.2 describes a trust web graph. In this figure, gray nodes are uncertain, and white nodes are seed vertices. v_i represents p_i. Here, we need to know if p_7 is credible or uncertain. According to the algorithm 1, first, the credibility scores of vertices which are pointed by seed vertices are worked out. We get $P_{v_4} = 1$, $P_{v_5} = 1$, and $P_{v_6} = 0.6$; then, according to the credibility scores of v_4, v_5 and v_6, we get $P_{v_7} = 1$; at last, if δ is set to 0.6, the page represented by v_7 is credible.

Algorithm 1. Naive algorithm

    Input : $TG(V,E,T),\delta,p_o,v_o$
    Output: the credibility-status of p_o
1 $S \leftarrow \Phi$;
2 **for** *each $v_i \in V$* **do**
3   **if** *v_i is seed vertex* **then**
4        $P_{v_i} = 1; S \leftarrow v_i$;
5   **else**
6        $P_{v_i}=0$;

7 **while** $v_o \notin S$ **do**
8   **for** *each $v_a \notin S \wedge v_a \in V$* **do**
9   **for** *each $v_i \in S$* **do**
10   **if** *v_i points to v_a* **then**
11            $C_{v_i,v_a} = P_{v_i} * t_{i,a}\ P_{v_a} = P_{v_a} + C_{v_i,v_a}$;
12   **if** $P_{v_a} \geqslant 1$ **then**
13               $P_{v_a} = 1$;

14        $S \leftarrow v_a$;

15 **if** $P_{v_o} \geqslant \delta$ **then**
16   **return** p_o is credible;
17 **else**
18   **return** p_o is uncertain;

FindCredPg. Since the trust value is less than 1. Given a path: $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k(3 \leqslant k)$. k is the length of the path. Only considering this path, P_{v_k} is related to k. If the k is enough greater, P_{v_k} may be less than δ. If the v_k is credible, the result is incorrect. In other cases, if a node v_o is pointed by $v_1, v_2 \ldots, v_k(3 \leqslant k)$, and $P_{v_i}(1 \leqslant i \leqslant k)$ is lower. According equation 2, P_{v_o} may be high. So, if the length of a path from a seed vertex to objective vertex is too long, the result is not accurate; if the paths from seed vertices to objective vertex are few, the result is not accurate; if the objective vertex are pointed by many vertices whose credibility scores are lower, the result is not accurate too. Just for reducing the impact of path length, number of paths and number of low credibility scores vertices, we improve the naive method and propose an another
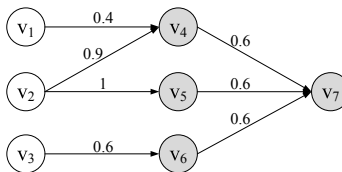

Fig. 2. A trust web graph

method: FindCredPg. The equation for credibility scores of a page in FindCredPg is below:

$$P_{v_o} = \begin{cases} 0 & \text{if } \sum_{i=1}^{n} P_{v_i} * t_{i,o} \lneq \delta \\ \\ 1 & \text{if } \sum_{i=1}^{n} P_{v_i} * t_{i,o} \geq \delta \end{cases} \tag{3}$$

In equation 3, v_o is pointed by $v_1, v_2, \ldots, v_n$ and δ is less than 1. Given a long path: $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k (3 \leqslant k)$, if $P_{v_i} = 1 (2 \leqslant i \leqslant k)$, the contribution to P_{v_o} from $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k (3 \leqslant k)$ is the same as $v_i \rightarrow \cdots \rightarrow v_k$. It means the impact of long paths is reduced. In addition, if a vertex is pointed by many vertices. Some vertices' contributions is deleted, since their credibility scores are too low. It means the impact of low credibility scores vertices is reduced. Thus, FindCredPg can bring more accurate result. As for the number of paths, it is determined by the seed set size.

5 Experiments

We run a set of experiments to compare naive method and FindCredPg in terms of accuracy and analyze the efficiency of FindCredPg. Firstly, we crawl a large set of pages from internet as data set. Secondly, we choose some credible pages from the data set as seeds. Then, some credible and uncertain pages are picked up as test data set. At last, we evaluate both methods in terms of accuracy under different seed set sizes and different values of δ. In addition, we analyze the efficiency of FindCredPg when the seed set size changes.

5.1 Experimental Setting

We construct a data set for this studying by crawling pages related to health in Chinese sites. Since uncertain pages often occur in health. We choose some sites involving health field as crawling seeds from open directory project.[1] Our data set is composed of pages and links. In this data set, 1.12 millions of pages and more than 14 millions of links are included. 67.3% pages contain keywords, while 88.1% pages contain titles. 6.1% pages don't contain keywords or title. More than 10% links are inter site links. Since, we focus on inter site links, other links are ignored. According to the pages and inter site links, we construct a trust web graph.

A set of credible pages are chosen as seeds. Usually Government sites, university sites and organization sites are considered credible. These pages from them are the candidate pages of seed set. Pages of other sites ranked in high position in Alexa,[2] are picked out as candidate pages, such as *www.39.net*. About 31% pages in data set are candidate pages. We use *SCS* to denote the seed candidate set which contains these candidate pages. The seed set is composed of the pages chosen from *SCS*.

---

[1] http://www.dmoz.org
[2] http://www.alexa.com

A set of pages are chosen as the test data set which is composed of credible pages and uncertain pages. These pages in test data set do not belong to *SCS*. First, we give some heuristic rules to find some credible page candidates, according to features mentioned in [4]. Then, if these pages involve health, we use Hon[1] to check them. At last,11 experienced users, who are graduate students and have used internet for a long time, help pick and label credible pages from candidates. In addition, some uncertain pages, which can not be proved credible because of lack of sufficient evidence, are labeled and included in the test data, too. At last, 500 credible pages and 500 uncertain pages are chosen as test data.

We adopt accuracy as measurement to evaluate our methods. The accuracy is defined as follows:

$$Accuracy = \frac{\text{Number of correct judgements}}{\text{Number of test data}} \tag{4}$$

Correct judgement means a credible page is considered credible and an uncertain page is considered uncertain by our methods.

In experiments, seed set size's range is from 20% to 100% of *SCS*. Since the content correlation values of pages are usually less than 0.8, the δ's range is from 0.1 to 0.7. The experiments run on an Intel Core 2 Quad 2.66Hz, windows 7 machine equipped with 4GB main memory.

5.2   Evaluation

Since web link structure is indispensable in the two methods. If the seed set size is small, the paths from seeds to an objective page are few and the result is influenced. Moreover, if the seed set size is too small, the paths may be missing. So, the size impact the accuracy heavily. In addition, the size influence the efficiency because the size influence the path length and the number of paths. δ is another important factor affecting accuracy. In naive method, δ is used in the last step. However, in FindCredPg, during the whole process, δ plays an important role.

Since our test data set is composed of credible pages and uncertain pages, we procedure experiments to evaluate the misjudgement ratio. Fig.3 shows the average misjudgement ratio on δ when proportion of *SCS* changes. For naive method, we use mcN
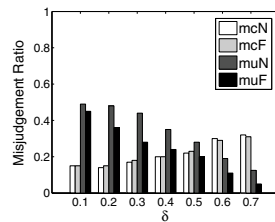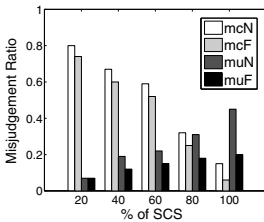

Fig. 3. Misjudgement ratio on various proportion of *SCS*

Fig. 4. Misjudgement ratio on various δ value

---

[1] https://www.hon.ch

and muN denote the misjudgement ratio on credible pages and uncertain pages respectively. For FindCredPg, mcF and muF are used. In Fig.3, as the seed set size increases, mcN and mcF become lower and lower. The reason is that the grater the seed set size, more and shorter paths from seed pages to objective pages are found and the credibility scores of credible pages are enhanced. On the contrary, muN and muF become higher and higher. Since more paths are found, the credibility scores of uncertain pages are enhanced and some uncertain pages are considered credible. In FindCredPg some pages' contributions to objective pages are filtered, so muF is lower than muN and mcF is lower than mcN.

Fig.4 describes the average misjudgement ratio on proportion of *SCS* when δ changes. With the increase of δ, mcN and mcF are enhanced. Since if the δ is greater, these credible pages, whose credibility scores are lower than δ, are regarded as uncertain. On the contrary, if the δ is smaller, some uncertain pages are considered credible, so muN and muF are reduced. In this figure, since FindCredPg reduces the impact of these lower credibility scores pages, muF is always smaller than muN. For the same reason, mcF is smaller than mcN. Fig.3 and Fig.4 show the advantage of FindCredPg in terms of misjudgement ratio.
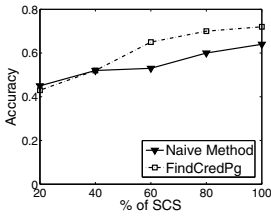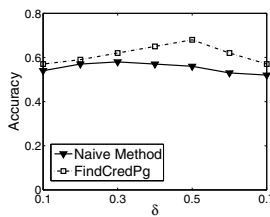

Fig. 5. Accuracy on various proportion of *SCS*

Fig. 6. Accuracy on various δ value
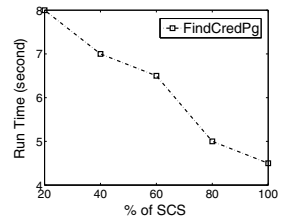
Fig. 7. Runtime on various proportion of *SCS*

The result we report in Fig.5 is average accuracy value on δ when proportion of SCS changes. Since the greater the proportion of *SCS* is, more and shorter paths from seeds to object vertex can be found. For naive method, with the increase in size of seed set, the accuracy is enhanced. When the size of seed set is 100%, the accuracy reaches peak. For FindCredPg, the trend is similar to naive method. When the size is 20%, the accuracy of naive method is more than the accuracy of FindCredPg. The reason is that for FindCredPg, contributions of these pages, whose credibility scores are lower, are filtered. Because of reducing the influence of long paths and lower credibility scores pages, when the size is greater than 20%, FindCredPg bring higher accuracy.

Fig.6 shows the average accuracy value on proportion of *SCS* when δ changes. For naive method, the accuracy increases and then decreases, with the increase of δ. The peak appears at $\delta = 0.3$. For FindCredPg, the trend is similar to naive method. Since the influence of long paths and some pages is reduced, accuracy of FindCredPg reaches peak(0.78) at $\delta = 0.5$. In addition. This figure reflects the superiority of FindCredPg in accuracy.

Fig.7 describes the average runtime on δ when proportion of *SCS* changes. Especially, this figure is about FindCredPg only, since there is no difference on efficiency between the two methods. The runtime is the time spent on assessing an objective page. Form section 4, we know if the objective page is close to seed pages, the efficiency is very high. On the contrary, the time spent on computing is more. So, when the seed set size is greater, more and shorter paths from seed pages to objective pages are found and the efficiency is higher. When the seed set size is 100% of *SCS*, the consuming time is the lowest. From our experiments result, the average runtime falls in the range from 2 to 8 seconds. In fact, we can make this method more efficient. For example, when assessing a page, we can find the paths from seed pages to the objective page and process the vertices on these paths only, which will reduce the computational complexity and bring higher efficiency.
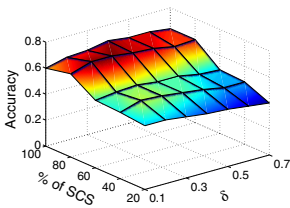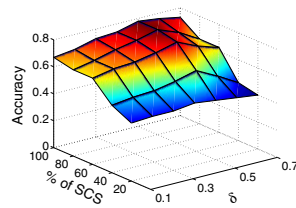

Fig. 8. Accuracy of naive method   **Fig. 9.** Accuracy of FindCredPg method

Fig.8 shows how the accuracy of naive method changes, considering both seed set size and δ. From this figure, when the size is less than 80% of SCS, the accuracy is enhanced with the increase of seed set size quickly. But, when the size is greater than 80%, the accuracy increases slowly. It means the most paths from seed pages to objective pages are found when the size is 80% of *SCS*. When $0.3 \leqslant \delta \leqslant 0.6$, the accuracy is higher. When $\delta \leqslant 0.3$, the accuracy is enhanced with the increase of δ. When $\delta \geqslant 0.3$, the accuracy decreases slowly. The reason is that when δ is minimal, some uncertain pages are considered as credible pages. When δ is greater, some credible pages are considered uncertain.

Fig.9 describes changes in accuracy of FindCredPg, considering seed set size and δ. It is similar to naive method, when the seed set size is about 80%, the accuracy is higher. When the size is more than 80% and $0.4 \leqslant \delta \leqslant 0.6$, the accuracy is higher. Comparing naive method, the peak(0.78) appears at $\delta = 0.5$ rather than $\delta = 0.3$. The reasons is follows: first, some pages' contributions to objective pages are filtered; second,the credibility scores of most pages in test data set fall in the range from 0.4 to 0.7. From Fig.8 and Fig.9, FindCredPg is more accurate. From these experiments, FindCredPg is superior in terms of accuracy. Its accuracy is near 80% and its efficiency is accepted.

6   Conclusion and Future Work

In this studying, we propose a way to construct a trust web graph based on the web link structure and the content correlation between pages. Based on the graph, FindCredPg

is proposed to asses pages. Although the trust relation is defined, other information, which may be helpful to measure trust relation, is ignored, because they are elusive. In addition, the efficiency of this method can be enhanced. In future work, we will try to find other information about measuring trust relation, enhance the efficiency of the method, and make our method more accurate and useful.

References

1. Gartner Puts Phishing Tab at \$3.2 Billion, `http://www.zdnet.com`
2. Gyongyi, Z., Garcia-Molina, G., Pedersen, J.: Combating Web Spam with TrustRank. In: 30th International Conference on Very Large Data Bases, pp. 576–587. Morgan Kaufmann Publishers, San Fransisco (2004)
3. Caverlee, J., Liu, L.: Countering Web Spam with Credibility-Based Link Analysis. In: 26th Annual ACM Symposium on Principles of Distributed Computing, pp. 157–166. ACM Press, New York (2007)
4. Fogg, B.J., Marshall, J., Laraki, O., Osipovich, A., Varma, C., Fang, N., Paul, J., Rangnekar, A., Shon, J., Swani, P., Treinen, M.: What makes Web sites credible?: A report on a large quantitative study. In: The CHI 2001 Conference on Human Factors in Computing Systems, pp. 61–68. ACM Press, New York (2001)
5. Schwarz, J., Morris, M.R.: Augmenting Web Pages and Search Results to Support Credibility Assessment. In: The International Conference on Human Factors in Computing Systems, CHI 2011, pp. 1245–1254. ACM Press, New York (2011)
6. Lucassen, T., Schraagen, J.M.: Trust in Wikipedia: How Users Trust Information from an Unknown Source. In: 4th ACM Workshop on Information Credibility on the Web, pp. 19–26. ACM Press, New York (2010)
7. Li, X., Meng, W., Yu, C.: T-verifier: Verifying Truthfulness of Fact Statements. In: 27th International Conference on Data Engineering, pp. 63–74. IEEE Press, New York (2011)
8. Yamamoto, Y., Tezuka, T., Jatowt, A., Tanaka, K.: Honto? Search: Estimating Trustworthiness of Web Information by Search Results Aggregation and Temporal Analysis. In: Dong, G., Lin, X., Wang, W., Yang, Y., Yu, J.X. (eds.) APWeb/WAIM 2007. LNCS, vol. 4505, pp. 253–264. Springer, Heidelberg (2007)
9. Nagura, R., Seki, Y., Kando, N., Aono, M.: A Method of Rating the Credibility of News Documents on the Web. In: 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 683–684. ACM Press, New York (2006)
10. Lee, R., Kitayama, D., Sumiya, K.: Web-based Evidence Excavation to Explore the Authenticity of Local Events. In: 2nd ACM Workshop on Information Credibility on the Web, pp. 63–66. ACM Press, New York (2008)
11. Urvoy, T., Chauveau, E., Filoche, P., Lavergne, T.: Tracking Web Spam with HTML Style Similarities. J. ACM Transactions on the Web 2(1) (2008)

Multiple Time Series Anomaly Detection
Based on Compression and Correlation
Analysis: A Medical Surveillance Case Study[*]

Zhi Qiao[1,2], Jing He[1], Jie Cao[3], Guangyan Huang[1], and Peng Zhang[2]

[1] Victoria University, Melbourne, Australia
[2] Institute of Information Engineering, Chinese Academic of Science
[3] Nanjing University of Finance and Economics, Nanjing, China
zhiqiao.ict@gmail.com, {Jing.He,Guangyan.Huang}@vu.edu.au
zhangpeng@ict.ac.cn

Abstract. In this paper, we present a novel anomaly detection framework for multiple heterogeneous yet correlated time series, such as the medical surveillance series data. In our framework, we propose an anomaly detection algorithm from the viewpoint of trend and correlation analysis. Moreover, to efficiently process huge amount of observed time series, a new clustering-based compression method is proposed. Experimental results indicate that our framework is more effective and efficient than its peers.

Keywords: multiple time series, anomaly detection, correlation.

1 Introduction

Anomaly detection is an important problem with wide applications [2, 3, 4, 6]. It aims to finding special patterns in data that do not conform to expected behaviours [1]. Existing methods mostly focus on detecting unusual subseries or outlier nodes from univariate time series. However, with the development of modern information systems, detecting anomaly from multiple correlated time series data attracts a large number of attentions recently.

Taking the medical application for example, anomaly detection technologies are essential to estimate physical condition of patients, as well as to avoid medical negligence. In surgery, doctors often use many different kinds of medical monitors to inspect a patient's current physiological status. Take the Intensive Care Unit (ICU) for example in Fig. 1. Doctors usually get samples from three kinds of respiratory data, Endtidal O2, FIO2 and bronchoalveolar lavage, both blocks with blue border indicate anomaly of physical status. Obviously, when anomaly occurs, time series will have significant pattern changes. Latent unknown relationship exists between pattern of

---

time series segment set and physical status of patients. Correlations among multiple observed time series are all used to indirectly indicate physical status of patients. However, each time series represents different vital signal.

Compared to existing anomaly detection methods, anomaly detection from correlated heterogeneous time series faces extra challenges: (1) heterogeneous time series from different monitors; (2) multiple time series are highly correlated because all data are captured from the same patient; (3) the relationship among multiple time series is latent and unknown.

Many existing methods can be used to detect anomaly from multiple time series. For example, method in [6] was used to detect abnormal trend evolution from multiple data streams. However, this method doesn't consider the case with huge amount of time series. Spiros Papadimitriou in [7] introduced SPIRIT, a streaming pattern discovery method in multiple time series data streams. In [8], Philip K. Chan proposed a method to model multiple time series. Both methods can be applied to mine multiple homogeneous time series; but it's inappropriate to directly mine multiple original series in different spaces. Existing works can't well address anomaly detection based on huge amount of multiple correlated heterogeneous time series.

Generally, anomaly of multiple time series data consists two parts: anomaly on each time series, and anomaly across multiple time series. Hence, we propose a framework for anomaly detection from multiple heterogeneous correlated time series, which can detect both anomaly on single time series and anomaly across multiple time series.

In our framework, we propose a novel anomaly detection algorithm based on multiple heterogeneous correlated time series from viewpoint of trend and correlation. Particularly, to properly process multiple time series when time series data evolve severely, we develop a clustering-based compression method. Anomaly measurement based on anomaly score is proposed to represent the deviation of the data from common patterns.

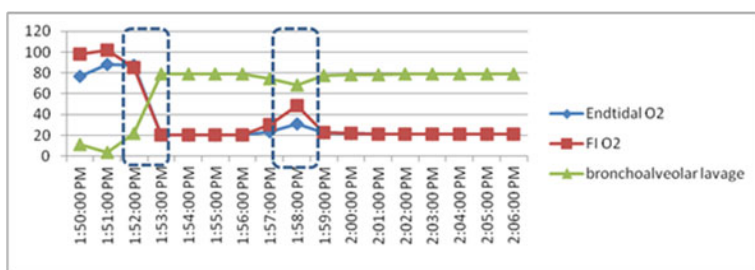**Fig. 1.** Real medical case

The rest of the paper is structured as follows. Section 2 introduces the proposed anomaly detection framework. Section 3 defines the multiple time series anomaly detection problem. Section 4 provides an algorithm to detect anomaly across multiple time series. Section 5 reports experimental results. Section 6 surveys the related work. We conclude the paper in Section 7.

2   Anomaly Detection Framework

Fig. 2 shows an overview of our framework for multiple time series anomaly detection. It is basically composed of three modules that we introduce as follows. The first module is a retrieve-unit, which use sensors to capture raw data and transport data to the process-unit, The second module is a process-unit, which checks each time series to detect anomaly according to domain knowledge. Especially, when an upcoming item does not comply with domain knowledge, the system sends an alarm to the alarm-unit. In domain knowledge, rules contain span of available value, span of changing volume and the correlation between the single value such as increasing of body temperature and decreasing of blood pressure. Another MTSAD algorithm is also proposed in case that multiple time series are correlated and heterogeneous. The last module is a alarm-unit. In the section followed, we mainly introduce the MTSAD algorithm in the framework.
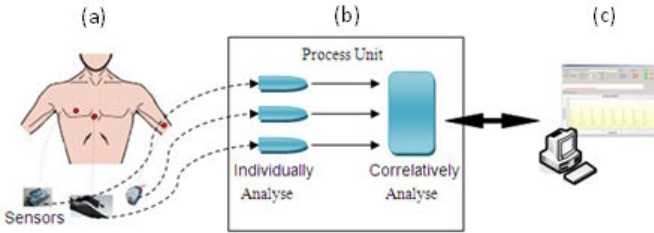

Fig. 2. Anomaly Detection Framework

3   Problem Settings and Outline of Proposed Algorithm

3.1   Problem Settings

Formally, we consider a monitor system consisting of n sensors. Each sensor samples a value once in a fixed interval and thus we can observe a series of this value's temporal varieties in a static window. Every k observation value can be seen as a time span along time series. k is predefined by domain knowledge. $S_t = \{ S_{t_1}, S_{t_2}, \ldots S_{t_n} \}$ represents a data sample consisting of n time series segments in t^{th} time span. Thus, $S_{ti} = \{ S_{t_{i1}}, S_{t_{i2}}, \ldots S_{t_{ik}} \}$, where S_{jt_i} represents the observation of the ith sensor at the jth time step in t^{th} time span ($i=1, 2, \ldots, n$; $j=1, 2, \ldots, k$; $t=1, 2, \ldots$). In the following, we introduce the MTSAD algorithm.

3.2   Outline of Proposed Algorithm

Algorithm 1 shows the outline of our proposed algorithm MTSAD. We first transform series from heterogeneous time series space into homogeneous trend space

by mapping original time-series segment into trend synopses sequence. Secondly, we compress data in order to reduce time cost of following operations. A compression algorithm is proposed to transform correlation matrix of multiple time series into correlation matrix of multiple clusters. We can use anomaly score of current time series set by employing multivariate normal distributions to reflect anomaly condition. In this way, we can detect anomaly by analyzing variation of anomaly score.

Algorithm 1:   MTSAD algorithm

Input: time series segment set S_t at t-th time span

Output: whether anomaly or not

1. M = H(S_t); //mapping time series in S_t from original space to trend space

2. Cluster series in *M* to get matrix *V* used to compress multiple series;

3. O = (VH(S_t))^T(VH(S_t))/(l-1); //Get correlation matrix *O*

4. Z_t = eigenvalues of matrix *O*;

5. Supposing Z_t conform to normal distribution, compute   p(Z_t);

6. $\tilde{s}_t$ = -log(p(Z_t)); //Compute anomaly score

7. J_t= $\tilde{s}_t$ - $\tilde{s}_{t-1}$; //Compute variation of anomaly score

8. Get changing degree of anomaly score α;

9. If α > predefined value   then Outcome=anomaly   Else Outcome=normal;

10. Return outcome

4   Algorithms for Anomaly Detection

4.1   Trend Synopses

We use the Haar wavelet transform to assess trend of time series segment [9]. Generally, the wavelet decomposition consists of an approximation of the detail coefficients that represent the function at various scales. The Haar wavelet is the simplest wavelet.

Fig. 3 presents decomposing procedure and relation between Haar wavelet with trend of series. After Haar wavelet transform, we can get the updated series **X**'{5; 6; 7; 4; 1; 8; 4; 2}. In **X**', the 2nd value, 6, can be computed from another prospective, [(19+17+12-4)-(7-1-3-7)]/23 as well. This is, sum of leaf node value of left sub-tree (19+17+12-4) minus sum of leaf node value of right sub-tree (7-1-3-7) then divide the *x*-th power of 2, where *x* represents layer in decomposing tree, where current node 6 is located. Apparently, it loosely presents that the trend is decreasing from the front half

part of series **X** to the back half part. The same analysis can be implemented for rest value of series **X**'. Hence **X**' can approximately synopses trend of **X**. Haar wavelet can be used to map X to **X**'. At last, we construct function H mapping series from original space to trend space, **X**'=H(**X**).

X' {5; 6; 7; 4; 1; 8; 4; 2}

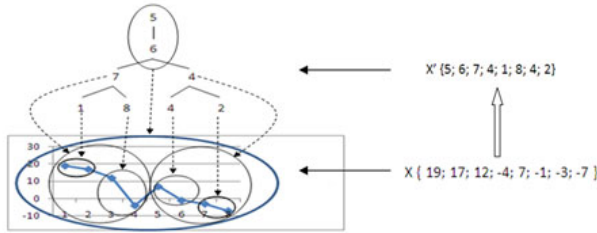X { 19; 17; 12; -4; 7; -1; -3; -7 }

Fig. 3. Relationship between Haar wavelet decomposition with trend of series

4.2   Feature Selection

We consider the time series segment set $S_t = \{ S_{t_1}, S_{t_2}, \ldots S_{t_n} \}$. Firstly, in order to handle the intense correlation among sensors, we construct a matrix C, whose components represent correlation strength between a pair time series.

$$C_{ij} = \frac{1}{l} \sum_{a=1}^{l} (S_{ti}(a) - \mu_{ti})(S_{tj}(a) - \mu_{tj}). \tag{1}$$

$$\mu_{ti} = \frac{1}{l} \sum_{a=1}^{l} S_{ti}(a). \tag{2}$$

$$\overset{\circ}{C} = \frac{1}{l-1} \sum_{a=1}^{l} S_{ti}(a) S_{tj}(a). \tag{3}$$

While C_{ij} represents component of matrix **C** at i-th row and j-th column at t-th time span, μ_{ti} represents the sliding window average of time series S_{ti} at t-th time span with the window size l. Thus, we call matrix **C** correlation strength matrix. Equation (3) can be used to unbiased estimate correlation strength matrix **C**. Formerly we have introduced trend synopses method based on Haar wavelet. Hence we use function H to map time series from original individual space to same trend space. Equation (4) is derived.

$$W = (H(S))^T (H(S)) / (l-1). \tag{4}$$

Then, we can get a vector **Z** consisting of eigenvalues of **W**. **Z** can be considered as feature of current time series segment set.

4.3   Compression

With the advance of sensor technology, more and more sensors are available to monitor the vital data in real time. As a result, we need to compress time series data in order to reduce time cost. Our clustering based compression method is summarized in Fig.4.
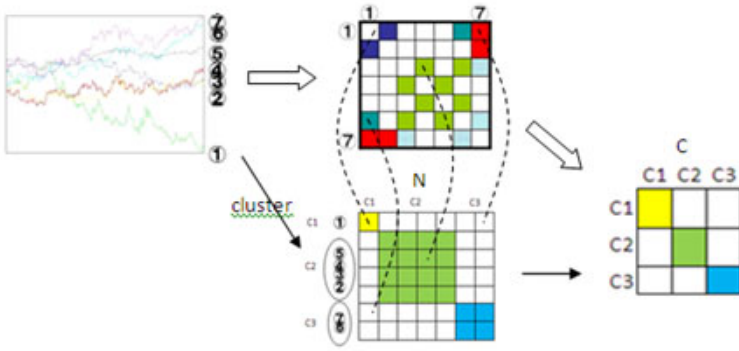

Fig. 4. Compression method based clustering

First, we use k-means algorithm to cluster n time series $\{ S_{t_1}, S_{t_2}, ... S_{t_n} \}$. These time series segments have been transformed by Haar wavelet and mapped into same trend space. So it can be used to measure distance between series. Then, we assume that a set of clusters $\{ C_k : k=1, ..., m \}$ is given and each series belongs to one of them. Consequently, each cluster consists of several similar series. Apparently, features of cluster are synopses of features of several similar time series belonging to this cluster. We can employ the correlation between clusters to estimate the correlation between time series.

Because correlation matrix **W** are derived based on sequence of time series. In order to implement projection from correlation between series to correlation between clusters, we define sequence of clusters according to sequence of observed time series. Scan sequentially series starting from the first series. When *i*-th time series is determined into *j*-cluster, if *j*-cluster hasn't existed in sequence of cluster, *j*-cluster is inserted into sequence. In this way, way can get non-repeated cluster sequence. When all of clusters have been insterted cluster sequence, scanning will be ended. Hence correlation matrix **O** between clusters can be derived based on cluster sequence.

$$W_{ij} \cong O_{C(i)C(j)} . \tag{5}$$

While W_{ij} represents the correlation between series i and j. $O_{C(i)C(j)}$ represents correlation between cluster $C(i)$ and $C(j)$. Here $C(i)$ represents a cluster which series *i* belongs to. We define loss function f

$$f_{ij} = (W_{ij} - O_{C(i)C(j)})^2 . \tag{6}$$

In order to minimize loss and improve accuracy of estimation, an estimation model is denoted in (7).

$$arg\min_{O_{ab}(a,b\in clustersets)}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(f_{ij}\right)^{2}. \tag{7}$$

In (6), our objective is to get matrix $\mathbf{O}$ to minimize difference between real value and the estimated value. In order to get optimal solution $\mathbf{O}$, we can equivalently transform (7) to (8).

$$arg\min_{O_{ab}(a,b\in clustersets)}\sum_{i=1}^{m}\sum_{j=1}^{m}\left[\sum_{p\in C(i)}\sum_{q\in C(j)}\left(W_{ij}-O_{pq}\right)^{2}\right]. \tag{8}$$

$$O_{pq}=\frac{\sum_{i\in cluster:p}\sum_{j\in cluster:q}W_{ij}}{amount(p)*amount(q)}. \tag{9}$$

Because the objective function in (8) is convex, we can get optimal solution by computing derivative. Equation (9) shows the solution of (8). In (9), "amout(p)" represents amount of time series in p-th cluster. In equation (3), we use $\mathbf{S_i}^{T}*\mathbf{S_j}/(l\text{-}1)$ to unbiasedly estimate W_{ij}. Therefore, equation (11) is derived from equation (9) and (10). Equation (11) implies that average of several time series belonging to a cluster can be used to synopses feature of the cluster and minimize loss of estimation.

$$\sum_{i\in cluster:p}\sum_{j\in cluster:q}W_{ij}=\sum_{i\in cluster:p}\sum_{j\in cluster:q}\frac{1}{l-1}(S_{i}^{T}S_{j})$$

$$=\frac{1}{l-1}\sum_{i\in cluster:p}S_{i}^{T}\sum_{j\in cluster:q}S_{j} \tag{10}$$

$$=\frac{1}{l-1}(\sum_{i\in cluster:p}S_{i})^{T}(\sum_{j\in cluster:q}S_{j}).$$

In order to compute optimal matrix $\mathbf{O}$, we define matrix $\mathbf{V}$ consisting of n vectors $\{\mathbf{V_1}\ \mathbf{V_2}\ \ldots\ \mathbf{V_n}\}$ and each vector with m dimensions is used to represent projection from time series to cluster. If j-th series belongs to i-th cluster, V_{ji} is 1/amount(i). If j-th series doesn't belong to i-th cluster, V_{ji} is 0. As a result, we can get optimal matrix $\mathbf{O}$ by equation (12) below. In equation (11), H is mapping function from original space to trend space, which has been introduced in 4.1.

$$O_{pq} = \frac{\frac{1}{l-1}(\sum_{i \in cluster:p} S_i)^T (\sum_{j \in cluster:q} S_j)}{amount(p) * amount(q)} \tag{11}$$

$$= \frac{1}{l-1}(\sum_{i \in cluster:p} S_i / amount(p))^T (\sum_{j \in cluster:q} S_j / amount(q)) \ .$$

$$O = (VH(S))^T (VH(S)) / (l-1) . \tag{12}$$

Apparently, size of $\mathbf{O}_{mxm}$ is less than $\mathbf{W}_{nxn}$. After getting compressed correlation matrix $\mathbf{O}$, vector $\mathbf{Z}_t$ consisting of eigenvalues of matrix $\mathbf{O}$ is obtained and can be considered as feature of current time series segment set S_t.

4.4   Anomaly Detection

When we have the feature vector $\mathbf{Z}_t$, we can score the anomaly in each time span. The scores represent how significantly data deviates from the learned normal patterns. Thus, high scores indicate anomalous data with low probability. According to central limit theorem, we suppose that $\mathbf{Z}_t$ conforms to normal distribution. We employ multivariate normal distributions for representing probability distributions p($\mathbf{Z}_t$) of $\mathbf{Z}_t$. We can estimate probability of current feature vector $\mathbf{Z}_t$ by equation (13). Then we can get anomalousness score by equation (14). In equation (13) and (14), μ_t and Σ_t represent exception and covariate matrix of feature vector $\mathbf{Z}_t$ respectively.

$$p(Z_t) = \frac{1}{(2\pi)^{m/2} |\Sigma_{t-1}|^{1/2}} \left[-\frac{1}{2}(Z_t - \mu_{t-1})^T \Sigma_{t-1}^{-1} (Z_t - \mu_{t-1}) \right] . \tag{13}$$

$$\tilde{s}_t = -\log p(Z_t) \ . \tag{14}$$

We estimate anomaly of current time series based on anomaly score from variation perspective. Considering continuity of time series and emergency of anomaly, variation of anomaly score can be used to detect anomaly better. We define J_t representing variation of anomaly score of current time series set. J_t can be obtained by equation (15).

$$J_t = \tilde{s}_t - \tilde{s}_{t-1} . \tag{15}$$

When we get variation of anomaly score of J_t, we compute changing degree. Changing degree is used to measure variation between J_{t-1} and J_t. Many methods can be used. Here we simply employ equation (16).

$$\alpha = | J_t / J_{t-1} | . \tag{16}$$

When changing degree α of current set exceeds a predefined value from domain knowledge, we can estimate that current set is anomaly.

5   Experiments

In this section, we present extensive experiments on real-world multiple time series data to validate performance and time cost of MTSAD algorithm. We compare our method with anomaly detection algorithm based on Eigen equation compression in [13]. We employ three groups of real-world datasets for evaluation. The experiments are conducted on a 3.0 GHZ CPU with 2 GB RAM and the experiment environment is windows XP with Matlab.

5.1   Performance

Medical Data. We used 11 medical time series[1]. Each time series was comprised of minutely data from starting anaesthesia 30/01/2007 11:28:00 AM to operation ending 4:03:00 PM. These data are generated from one unnamed Australian Hospital.

Results and Discussions. As a measure of the accuracies, we employed anomaly estimated precision (true positive rate) and anomaly recall rate. Recall rate is defined as the number of true positives divided by the total number of elements that actually belong to the positive class. Both true positive rate and recall rate are in [0, 1] and higher value corresponds to higher accuracy and better performance.

Table 1. Performance comparision

Operation	Algorithm in [13]		MTSAD algorithm	
	true positive rate	recall	true positive rate	recall
Non-Compressed	0.81	0.83	0.86	0.88
Compressed-4	0.51	0.57	0.51	0.58
Compressed-6	0.54	0.63	0.73	0.78
Compressed-8	0.6	0.71	0.8	0.86

We show the experimental results in table (1). We compared our proposed MTSAD algorithm with algorithm in [13]. From the results we can observe that MTSAD has better performance for anomaly detection.

Financial Data. Our proposed algorithm can be used to represent situation of national economic development. We use 11 economical time series consisting of GS1, DTB3, TB3MS, WTB3MS, GS5, GS10, MPRIME, WPRIME, FEDFUNDS, AAA, BAA respectively. Each time series was average of monthly data from January 1960 to January 2010. All of data are gotten from the website of Federal Reserve Bank of St. Louis[2]. We employed them for the following reasons. Firstly, financial data reflects

---

[1] Observed medical time series consist of respiratory data of patient in surgery after performing anaesthesia, Respiratory Rate, sevoflurane Endtidal, sevoflurane FI, minimum alveolar concentration, Endtidal CO_2, FI CO_2, Endtidal O_2, FI O_2, Endtidal NO_2, FI NO_2, bronchoalveolar lavage respectively in the experiment.

[2] Website URL of Federal Reserve Bank of St. Louis: http://research.stlouisfed.org/fred2/

situation of economic development. There is latent relation between recession with observed financial data. Secondly, there are strong correlations among financial time series.   Thirdly, the experimental results are intuitively interpretable.

Results and Discussions. In this experiment, we observe the behaviors of the proposed method only qualitatively to indirectly validate the performance of MTSAD algorithm compared with another anomaly detection method in [13] because we could not know what the anomalies were to be detected though we know there is unknown latent relationship between financial data and recession. Hence we only employ both algorithms to estimate anomaly score for time series with compression size 4. The anomalousness score of time series are shown in Figure 5. US recession date can be gotten from the website of National Bureau of Economic Research (NBER) indicated by shaded areas in Figure (5). In Figure 5, the green curve represents anomaly score gotten by anomaly detection algorithm in [13]. The red curve represents anomaly score gotten by our method. Shaded areas in Figure (5) indict US recession.   In figure (5), we see that several peaks appear near to recession date. Both curves can detect latent anomaly (recession) of financial time series. In red curve, peaks appear nearer to recession date than the green one. Apparently, fewer noises are raised and performance of anomaly detection is better. The experimental results indicate that our proposed method is applicable to detect latent anomaly from financial time series.
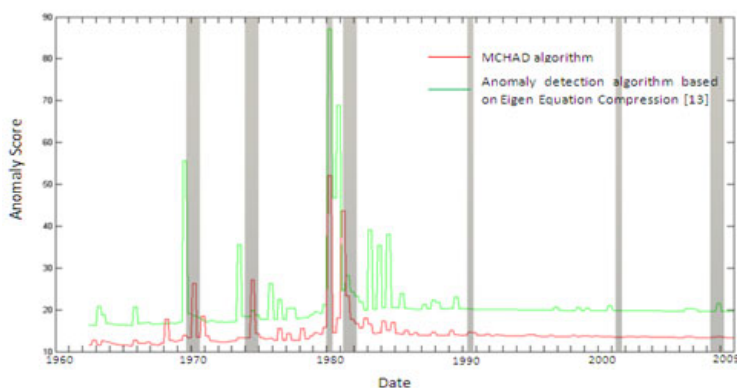
Fig. 5. Performance comparision: MTSAD algorithm VS [13]

5.2   Time Cost

Our proposed anomaly detection algorithm employs novel compression method based on clustering to reduce anomaly detection time. Advantage on time cost can be showed on huge scale of real time series. Real medical data set are still small scale.

Chlorine Dataset. It was generated by EPANET 2. Due to the page limit, please refer to [8] for detailed description. Amount of observed time series in Chlorine dataset, 166, is bigger than amount in medical set, 11. In the experiment, we just use part of time series in Chlorine dataset and random select 5, 10, 15, 20, 25, 30, 35 and 40 time series.

Results and Discussions. In this experiment, we observed anomaly detection time of the proposed method only qualitatively and compared time cost under different

compression size because we could not know what the anomalies were to be detected for Chlorine dataset. The experimental results are shown in figure (7). Apparently, as amount of observed time series increases, anomaly detection time cost increases. Time cost can be significantly reduced using our proposed compression method when amount of observed time series is given.
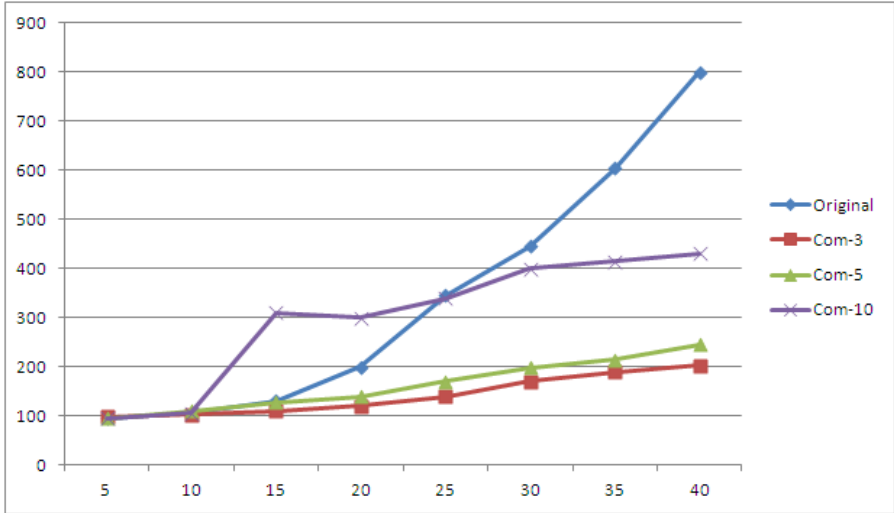

Fig. 6. Time cost vs. Amount of observed time series

6   Related Works

Anomaly detection techniques, particularly those based on statistical unsupervised learning, such as outlier detection [10, 12] and change-point detection [11, 14, 15], also can be applied to our problem. However, we desire to model multiple time series considering correlation and latent pattern significantly changing. There exist many methods for detecting anomaly over multiple time series [6, 7, 8]. However, they don't consider time series that are heterogeneous. Distance and correlation cannot be measured among time series in different original space. In [16], an abnormal patterns mining algorithm from heterogeneous time series was presented. However, irrelevant features supposed were different from multiple time series with strong correlations in our considering.

7   Conclusions

Time series anomaly detection has been widely studied. In this paper, we propose a new anomaly detection framework for multiple heterogeneous correlated time series. In the framework, we propose an anomaly detection algorithm based on multiple heterogeneous correlated time series from perspective of trend and correlation (MTSAD). In order to process on multiple time series when time series data evolve dramatically, a clustering-based compression method in MTSAD is proposed to reduce time cost. At last, we conduct extensive experiments on real world data sets to demonstrate the efficiency and effectiveness of our proposed algorithm.

The main contribution of this study is: (1) we consider correlation among multiple time series to mine anomaly. (2) We map heterogeneous space into trend space using novel trend synopsis method. (3) A clustering-based compression method is proposed.

References

1. Kriegel, H.-P., Kröger, P., Schubert, E., Zimek, A.: Outlier Detection in Axis-Parallel Sub-spaces of High Dimensional Data. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 831–838. Springer, Heidelberg (2009)
2. Patcha, A., Park, J.: An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer Networks 51, 3448–3470 (2007)
3. Duncan, G., Gorr, W., Szczypula, J.: Forecasting Analogous Time Series. Carnegie Mellon University, Pittsburgh, PA
4. Keogh, E., Lin, J., Fu, A.: HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM 2005 (2005)
5. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic Discovery of Time Series Motifs. In: Proceeding of SIGKDD 2003, Washington, DC, USA, August 24-27 (2003)
6. Zhang, C., Weng, N., Chang, J., Zhou, A.: Detecting Abnormal Trend Evolution over Multiple Data Streams. In: Li, Q., Feng, L., Pei, J., Wang, S.X., Zhou, X., Zhu, Q.-M. (eds.) APWeb/WAIM 2009. LNCS, vol. 5446, pp. 285–296. Springer, Heidelberg (2009)
7. Papadimitriou, S., Sun, J., Faloutsos, C.: Streaming Pattern Discovery in Multiple Time-Series. In: Proceedings of the 31st VLDB Conference, Trondheim, Norway (2005)
8. Chan, K.P., Mahoney, V.M.: Modeling Multiple Time Series for Anomaly Detection. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM 2005 (2005)
9. Craigmile, F.P., Guttorp, P., Percival, B.D.: Trend Assessment in a Long Memory Dependence Model using the Discrete Wavelet Transform. Environmetrics 15(4), 313–335
10. Breunig, M.M., Kriegel, P.H., Ng, T.R., Sander, J.: Lof: Indentifying densitybased local outliers. In: Proceeding of ACM SIGMOD Conference (2000)
11. Guralnik, V., Srivastava, J.: Event detection from time series data. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 33–42 (1999)
12. Ma, J., Perkins, S.: Time-series novelty detection using one-class support vector machines. In: Proceedings of International Joint Conference on Neural Networks (2003)
13. Hirose, S., Yamanishi, K., Nakata, T., Fujimaki, R.: Network Anomaly Detection based on Eigen Equation Compression. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Miing, KDD 2009 (2009)
14. Xuan, X., Murphy, K.: Modeling changing dependency structure in multivariate time series. In: Proceedings of the 24th International Conference on Machine Learning, pp. 1055–1062 (2007)
15. Takeuchi, J., Yamanishi, K.: A unifying framework for detecting outliers and change points from time series. IEEE Transactions on Knowledge and Data Engineering 18(4), 482–492 (2006)
16. Fujimaki, R., Nakata, T., Tsukahara, H., Sato, A., Yamanishi, K.: Mining Abnormal Patterns from Heterogeneous Time-Series with Irrelevant Features for Fault Event Detection. Statistical Analysis and Data Mining 2 (2009)

Energy-Based Metric for Ensemble Selection[*]

Weimei Zhi, Huaping Guo, and Ming Fan

School of Information Engineering, Zhengzhou University, P.R. China
{iewmzhi,mfan}@zzu.edu.cn, hpguo.gm@gmail.com

Abstract. Ensemble selection copes with the reduction of an ensemble of the predictive models to reduce its response time and increase its accuracy. A number of selection methods via greedy search of the space of all possible ensemble subsets have been recently proposed. The major issue of these algorithms is to construct an effective metric to supervise the search process. In this paper, we view the issue of ensemble problem from a new viewpoint: energy-based learning, and then contribute a novel metric called EBM (Energy-based Metric) to guide the search. Also, this metric takes into account the strength of the decision of the current ensemble. Empirical results show that using the proposed metric to select subensemble leads to significantly better accuracy results compared to state-of-the-art metrics.

Keywords: Ensemble Selection, Energy based Metric.

1 Introduction

Ensemble of multiple learning machines has been a very popular research topic during the last decade in machine learning and data mining. It is well accepted that an ensemble usually generalizes better than an individual classifier alone. The key to ensemble success is to create a collection of diverse and accurate base classifiers [1].

Many approaches have been proposed to create accurate and diverse ensemble members. Examples include bagging [2], boosting [3], random forest [4] and COPEN [5]. One problem existing in these approaches is that they tend to train a very large number of classifiers which increase the response time for prediction. Equally important is that they tend to construct an ensemble with both high and low accurate classifiers. Intuitively, the latter may affect the expected performance of the ensemble. Ensemble selection [6,7,8], also called ensemble pruning, is a technique to tackle these problems by selecting an optimal/suboptimal subset as subensemble for prediction.

Given an ensemble with M members, the problem of finding the subset of ensemble members with best performance involves searching the space of $M^2 - 1$ non-empty subensembles, which is proved to be an *NP*-complete problem [9]. Several efficient ensemble selection methods [10,11,12,13] via a greedy search policy have been proposed: given a subensemble S which is initialized to be empty

(full), searching for the space of different combination candidates by iteratively adding into (removing from) S the classifier $h \in H \backslash S$ ($h \in S$) that optimizes a cost function. The major issue in this approach is to design an effective metric to guide the search process.

In this paper, we first views ensemble selection from a new angle: energy-based learning [14], based on which a new metric called EBM (Energy-based Metric) is proposed to guide the search of ensemble selection. This metric also takes into account the strength of the decision of the current ensemble. Experimental results show that, compared with state-of-the-art metrics based ensemble selection algorithms, EBM based algorithm induces subensembles with significantly better performance.

The remainder of this paper is structured as follows: after reviewing the related work in section 2 and section 3, section 4 introduces the proposed metric. Section 5 presents the experimental results, and finally, section 6 concludes this work.

2   Metric for Greedy Ensemble Selection

The evaluation metric is the main component that differentiates greedy ensemble selection methods. Traditional valuation metrics can be grouped into two major categories: those that are based on performance and those on diversity.

2.1   Performance-Based Metric

The goal of performance-based metrics is to find the model that maximizes the performance of the ensemble produced by adding (removing) a model to (from) the current ensemble. Many performance-based metrics have been proposed, such as accuracy, root-mean-squared-error, mean cross-entropy, lift, precision/recall break-even point, precision/recall F-score, average precision and ROC area [15].

The time complexity of the calculation of performance-based measure is $O(|H|N)$, since the calculation requires the decision of current ensemble on each example of selection set, where $|H|$ is the size of current ensemble and N is the size of current validation set. We can optimize the complexity to be $O(N)$ if we update the current ensemble incrementally: only one classifier is added to (removed from) it for each iteration.

2.2   Diversity-Based Metric

Let h be a classifier and let H be an ensemble. Partalas et al. identified that the prediction of h and H on an example $\mathbf{x}_i$ can be categorized into four cases: (1) e_{tf}:$h(\mathbf{x}_i)=y_i \wedge H(\mathbf{x}_i)\neq y_i$, (2) e_{tt}:$h(\mathbf{x}_i)=y_i \wedge H(\mathbf{x}_i)= y_i$, (3) e_{ft}:$h(\mathbf{x}_i)\neq y_i \wedge H(\mathbf{x}_i)= y_i$, and (4) e_{ff}:$h(\mathbf{x}_i)\neq y_i \wedge H(\mathbf{x}_i)\neq y_i$, where y_i is the desired label associated with $\mathbf{x}_i$. They concluded that considering all four cases is crucial to design ensemble diversity metrics [12].

Many diversity metrics are designed by considering the four cases. Examples include the complementariness [10] and concurrency [11]. The complementariness of h with respect to H and a selection set D_s is calculated by

$$COM(h, H) = \sum_{\mathbf{x}_i \in D_s} I(\mathbf{x}_i \in e_{tf}), \tag{1}$$

where $I(true) = 1$, $I(false) = 0$. The complementariness is exactly the number of examples that are correctly classified by h and incorrectly classified by H. The concurrency is defined by

$$CON(h, H) = \sum_{\mathbf{x}_i \in D_s} (2I(\mathbf{x}_i \in e_{tf}) + I(\mathbf{x}_i \in e_{tt}) - 2I(\mathbf{x}_i \in e_{ff})), \tag{2}$$

which is similar to the complementariness, with the difference that it considers two more cases and weights them.

Unlike complementariness and concurrency, Partalas et al. [12,16] introduces a new metric called UWA (Uncertainty Weighted Accuracy) which considering all four cases discussed above. UWA is defined as

$$CON(h, H) = \sum_{\mathbf{x}_i \in D_s} (NT_i * I(\mathbf{x}_i \in e_{tf}) + NF_i * I(\mathbf{x}_i \in e_{tt})$$
$$- NF_i * I(\mathbf{x}_i \in e_{ft}) - NT_i * I(\mathbf{x}_i \in e_{ff})), \tag{3}$$

where NT_i is the proportion of classifiers in the current ensemble S that predict $\mathbf{x}_i$ correctly, and $NF_i = 1 - NT_i$ is the proportion of classifiers in S that predict $\mathbf{x}_i$ incorrectly. Besides considering all four cases, this metric take into account the strength of the decision of the current ensemble [12].

In this paper, we propose a novel metric to guide the search of greedy ensemble selection, which is designed based on Energy-based Learning (EBL). This metric also considers the strength of the decision of the current ensemble, while it reviews the strength from the EBL based angle. More details about EBL is discussed in next section.

3   Energy-Based Learning

This paper only focuses on classification problem. Without loss generalization, let us define our task as predicting the best configuration of label (variable) $y \in \Omega = [1, K]$, given a set of observed (input) variable collectively denoted by $\mathbf{x}$. An energy-based model [14] associates each configuration $(\mathbf{x}, y)$ with a scaler energy $E(\mathbf{w}, \mathbf{x}, y)$ where $\mathbf{w}$ is a parameter vector, which is to be learned. This energy can be viewed as the measure of "comparable" between $\mathbf{x}$ and y.

The inference is to search for the desired answer y in a set Ω that minimizes the energy. More precisely, the input $\mathbf{x}$ is given, and the model must produce the value y^*, chosen from the set Ω, for which $E(\mathbf{w}, \mathbf{x}, y)$ is the smallest:

$$y^* = \arg \min_{y : y \in \Omega} E(\mathbf{w}, \mathbf{x}, y). \tag{4}$$
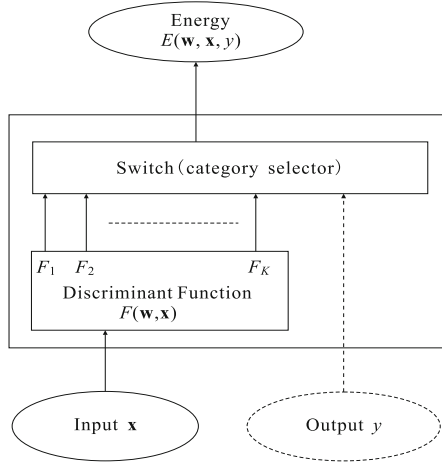
Fig. 1. Energy-based model for multi-class classifying problem

The architecture of energy-based model (for multi-class classifiers) is shown in Fig. 1. A parameterized discriminant function $F(\mathbf{w}, \mathbf{x})$ produces an output vector with one component for each of the K labels. Component F_k is interpreted as the energy (or penalty) for assigning $\mathbf{x}$ to the i-th category. A discrete switch module selects the component which is connected to the output energy. The position of the switch is controlled by the discrete y, which is interpreted as the category. The output energy is equal to $E(\mathbf{w}, \mathbf{x}, y) = \sum_{k=1}^{K} \delta(y - k) F(\mathbf{w}, \mathbf{x})_k$, where $\delta(y - k) = 1$ if $y = k$ and 0 otherwise, and $F(\mathbf{w}, \mathbf{x})_k$ is the k-th component of $F(\mathbf{w}, \mathbf{x})$. Running the machine consists in finding the position of the switch (the value of y) that minimizes the energy, i.e. the position of the switch that selects the smallest component of $F(\mathbf{w}, \mathbf{x})$.

3.1 Loss Function for Energy-Based Learning

Let $D_{tr} = \{\mathbf{x}_i | i = 1...N\}$ be an example set and let $y_i \in \Omega = \{1, 2, ..., K\}$ is the desired label associated with example $\mathbf{x}_i$. The major issue in energy-based learning is to design a suitable loss function and minimize the function. For a given example $\mathbf{x}_i$, the purpose of EBL is to shape the energy-based metric so that corresponding desired output configurations $(\mathbf{x}_i, y_i)$ have lower energy than other configurations.

For many cases, the loss function is defined as

$$\mathcal{L} = \frac{1}{N} \sum_{\mathbf{x}_i \in D_{tr}} L(\mathbf{w}, \mathbf{x}_i, y_i), \tag{5}$$

where $L(\mathbf{w}, \mathbf{x}_i, y_i)$ is the loss function for each example $\mathbf{x}_i$. Here, we assume that L depends on $\mathbf{x}_i$ only indirectly through the set of energies $\{E(\mathbf{w}, \mathbf{x}_i, y), y \in \Omega\}$, i.e. the per-example loss for example $(\mathbf{x}_i, y_i)$ is the following form:

$$L(\mathbf{w}, \mathbf{x}_i, y_i) = L(y_i, E(\mathbf{w}, \mathbf{x}_i, 1)...E(\mathbf{w}, \mathbf{x}_i, K)). \tag{6}$$

To correctly classify example $\mathbf{x}_i$, we must characterize the form that L can take such that minimization will eventually drive the machine to satisfy condition 1:

Condition 1. $E(\mathbf{w}, \mathbf{x}_i, y_i) < E(\mathbf{w}, \mathbf{x}_i, y)$, $\forall y \neq y_i$.

Let us denote by $\overline{y}$ the output that produces the smallest energy while being different from the desired output y_i, i.e., $\overline{y} = \arg\min_{y \in \Omega, y \neq y_i} E(\mathbf{w}, \mathbf{x}_i, y)$. Condition 1 can be rewritten as:

Condition 2. $E(\mathbf{w}, \mathbf{x}_i, y_i) < E(\mathbf{w}, \mathbf{x}_i, \overline{y})$, $\overline{y} = \arg\min_{y \in \Omega, y \neq y_i} E(\mathbf{w}, \mathbf{x}_i, y)$.

To ensure that the correct answer is robustly stable, we may choose to impose that the energy of the desired output be lower than the energies of the undesired outputs by a margin m:

Condition 3. $E(\mathbf{w}, \mathbf{x}_i, y_i) < E(\mathbf{w}, \mathbf{x}_i, \overline{y}) - m$, $\overline{y} = \arg\min_{y \in \Omega, y \neq y_i} E(\mathbf{w}, \mathbf{x}_i, y)$.

Therefore, we must design L in such a way that minimizing it will decrease the difference $E(\mathbf{w}, \mathbf{x}_i, y_i) - E(\mathbf{w}, \mathbf{x}_i, y)$, whenever $E(\mathbf{w}, \mathbf{x}_i, y_i) \geq E(w, \mathbf{x}_i, y) - m$. In other words, whenever the difference between the energy of the desired answer and the energy of the incorrect answer with the lowest energy is larger than a threshold (margin), our learning procedure should make the difference smaller.

In this paper, we apply energy-based learning to ensemble selection and then design a new metric (loss function) called EBM (Energy-based Metric) to supervise ensemble selection methods. This metric satisfies condition 3 and thus it guarantees that minimizing it will eventually drive the final selected subensemble correctly classifying a given example.

4 Energy-Based Metric for Ensemble Selection

In this section, we first view ensemble selection from a new viewpoint: Ensemble-based Learning, and then present the proposed metric for ensemble selection.

4.1 Energy-Based Learning for Ensemble Problem

Suppose that each member h in ensemble $H = \{h_t | t = 1, 2, ...M\}$ maps an example $\mathbf{x}_i$ to a label y, $h(\mathbf{x}_i) = y \in \Omega = [1, K]$. According to the discussion in section 3, the energy of configuration $(\mathbf{x}_i, y)$ and the prediction of ensemble on example $\mathbf{x}_i$ are defined as Eq. 7 and Eq. 8 respectively, where $w_t \in \{0, 1\}$ is the weight associated with classifier h_t. Obviously, the prediction of H on example $\mathbf{x}_i$ is the same as the prediction of H using weighted-majority voting.

$$E(\mathbf{w}, \mathbf{x}_i, y) = -\sum_t w_t \delta(h_t(\mathbf{x}_i) = y) \tag{7}$$

$$H(\mathbf{x}_i) = \arg\min_{y : y \in \Omega} E(\mathbf{w}, \mathbf{x}_i, y) \tag{8}$$

Based on Eq. 7 and Eq. 8, the problem of ensemble selection can be viewed as: setting most items in w to be **0** (others to be **1**) such that the corresponding energy loss function is minimized. This problem is *NP*-Complete which prohibits enumerate all setting candidates [9]. In terms of this problem, several efficient ensemble selection methods [10,11,12,13,16] via a greedy search policy have been proposed: starting with an initial vector **w** to be **0** (**1**), and then iteratively setting the item w_i in **w** to be 1 (0) that minimizes the loss function. This method can also be described as: starting with an empty (full) initial ensemble and searching the space of different ensembles by iteratively expanding (contracting) the initial ensemble by a single model.

This paper employs the above method to select optimal/suboptimal subensemble, though many EBL based approaches can be adopted, which is one of our future work.

4.2   Proposed Metric

In this section, we design a novel metric (EBM, Energy-based Metric) which considers the strength of the decision of current ensemble. Greedy ensemble selection approaches start either with the weight vector of classifiers to be **0** or **1** vector. For simplicity of presentation we focus on the former initial conditions only, yet our argumentation holds for both.

Based on the prediction of h on examples, the selection set D_s is grouped into two sets:

$$e_t = \{\mathbf{x}_j | \mathbf{x}_j \in D_s \wedge h(\mathbf{x}_j) = y_j\}$$
$$e_f = \{\mathbf{x}_j | \mathbf{x}_j \in D_s \wedge h(\mathbf{x}_j) \neq y_j\}. \tag{9}$$

The EBM (Energy-based Metric), the proposed loss function, of a classifier h with respect to ensemble S and selection set D_s is defined as

$$\mathcal{L} = \sum_{\mathbf{x}_i \in D_s} L(\mathbf{w}, \mathbf{x}_i, y_i), \tag{10}$$

where

$$L(\mathbf{w}, \mathbf{x}_i, y_i) = \frac{I(\mathbf{x}_i \in e_f) - I(\mathbf{x}_i \in e_t)}{|E(\mathbf{w}, \mathbf{x}_i, y_i) - E(\mathbf{w}, \mathbf{x}_i, \overline{y}) + m| + 1}, \tag{11}$$

where $m > 0$ is the margin as defined in condition 3, and the constant 1 is to avoid $|E(\mathbf{w}, \mathbf{x}_i, y_i) - E(\mathbf{w}, \mathbf{x}_i, \overline{y})| = 0$ being too small or zero. The specific analysis is as follows:

- If $\mathbf{x}_i \in e_t$, then $L(\mathbf{w}, \mathbf{x}_i, y_i) \geq 0$, since h correctly classifies $\mathbf{x}_i$ that make the energy of desired configuration $(\mathbf{x}_i, y_i)$ smaller. Otherwise, $L(\mathbf{w}, \mathbf{x}_i, y_i) \leq 0$, since h incorrectly classifies $\mathbf{x}_i$ that make the energy of undesired configuration $(\mathbf{x}_i, h(\mathbf{x}_i))$ smaller.
- From condition 3, we have that $|E(\mathbf{w}, \mathbf{x}_i, y_i) - E(\mathbf{w}, \mathbf{x}_i, \overline{y}) + m|$ is exactly the distance between the energy margin and the energy of correctly (incorrectly) classifying example $\mathbf{x}_i$ by H, and thus it reflects the confidence that $\mathbf{x}_i$ is correctly (incorrectly) classified by H. If $|E(\mathbf{w}, \mathbf{x}_i, y_i) - E(\mathbf{w}, \mathbf{x}_i, \overline{y}) + m|$ is

small or zero, $\mathbf{x}_i$ is on the margin that ensemble H correctly (incorrectly) classifies $\mathbf{x}_i$, and thus changing the weight of one classifier to be 1 may change ensemble prediction on the example. Therefore, $\frac{1}{|E(\mathbf{w},\mathbf{x}_i,y_i)-E(\mathbf{w},\mathbf{x}_i,\overline{y})+m|}$ is large, i.e., the impact of changing the weight of one classifier to be 1 is large. On the other hand, if $|E(\mathbf{w},\mathbf{x}_i,y_i)-E(\mathbf{w},\mathbf{x}_i,\overline{y})+m|$ is large, ensemble H correctly (incorrectly) predict $\mathbf{x}_i$ with high confidence, and thus changing the weight of one classifier to be 1 does not affect ensemble prediction a lot. Therefore, $\frac{1}{|E(\mathbf{w},\mathbf{x}_i,y_i)-E(\mathbf{w},\mathbf{x}_i,\overline{y})+m|}$ is small, i.e., the impact of setting one classifier weight to be 1 is small.

In this way, L (proposed per-example energy loss), and thus $\mathcal{L}$ (the proposed energy loss metric), considers the strength of current ensemble decision on each example.

For a particular example $\mathbf{x}_i$, the per-example loss L should be designed in such a way that its minimization with respect to $\mathbf{w}$ will make the energy of the correct answer lower than the energies of all possible incorrect answers. Minimizing such a loss function will make the machine produce the right answer when running the energy-minimizing inference procedure. Theorem 1 guarantees that Eq. 11 (the proposed per-example loss function) satisfies the property, if $L(\mathbf{w},\mathbf{x}_i,y_i) \leq 0$ and the number of classifiers in ensemble H that correctly classifies $\mathbf{x}_i$ is larger than the margin m.

Theorem 1. *Let $S \subseteq H$ be a subset of ensemble set H, and let each classifier $h \in S$ correctly classify example $\mathbf{x}_i$, i.e., $h(\mathbf{x}_i) = y_i$. For $\mathbf{x}_i$, minimizing Eq. 11 guarantees that condition 3 holds, if $|S| > m$ and $L(\mathbf{w},\mathbf{x}_i,y_i) \leq 0$.*

Proof. Greedy ensemble selection starts either with the weight vector of ensemble members to be $\mathbf{0}$ or be $\mathbf{1}$. We only focus on the former initial conditions for simplicity reason, yet our proof holds for both.

For configuration $(\mathbf{x}_i, y_i)$, repeating minimizing Eq. 11 will result in that, firstly the weight associated with each member $h \in S$ (correct classifier) is set to be 1 and then the weight associated with $h \in H\backslash S$ (incorrect classifier) to be 1. Since $L(\mathbf{w},\mathbf{x}_i,y_i) \leq 0$, the iteration stops until the former case occurs. Combining with Eq. 7, we have that

$$E(\mathbf{w},\mathbf{x}_i,y_i) = -\sum_t w_t \delta(h_t(\mathbf{x}_i) = y_i) = -|S| \tag{12}$$

$$E(\mathbf{w},\mathbf{x}_i,\overline{y}) = -\sum_t w_t \delta(h_t(\mathbf{x}_i) = \overline{y}) = 0, \tag{13}$$

where $\overline{y}$ is the output that produces the smallest energy while being different from the desired output y_i, as defined before. Since Eq. 12 and $|S| > m$,

$$E(\mathbf{w},\mathbf{x}_i,y_i) < E(\mathbf{w},\mathbf{x}_i,\overline{y}) - m. \tag{14}$$

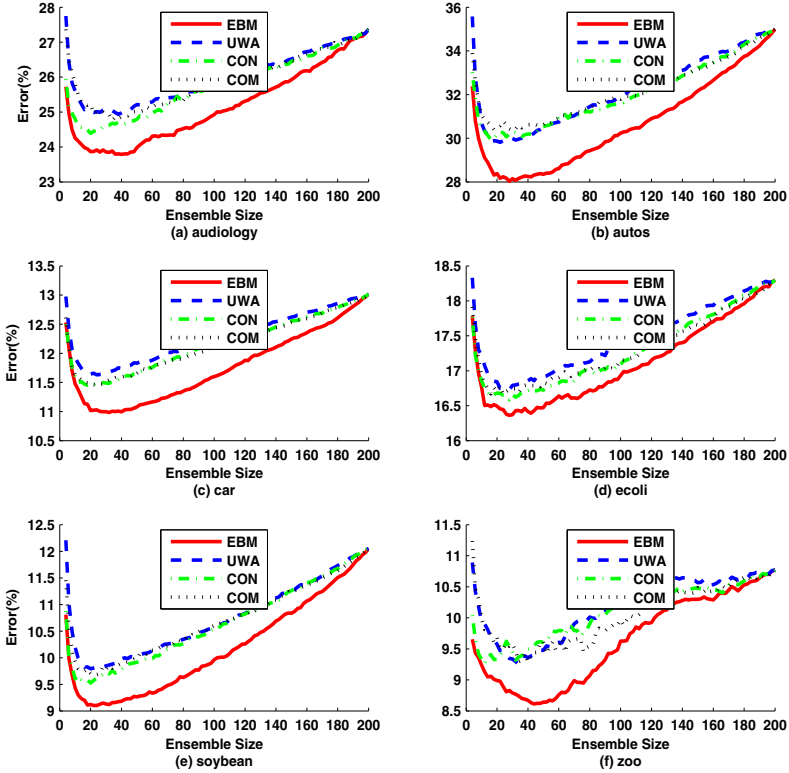Therefore Eq 14 holds. $\qquad\square$

Fig. 2. Comparative results for six data sets in the first category

5 Experiments

5.1 Data Sets and Experimental Setup

24 data sets, of which the characteristics are not shown due to space limit, are randomly selected from the UCI repository [17]. Each data set is randomly divided into three subsets with equal size, where one is used for training model, one for pruning model, and the other one for testing model. There are six permutations of the three subsets and thus experiments on each set consist of six sub-experiments. We repeat 50 independent trials on each data set. Therefore a total of 300 trials of experiments are conducted on each data set.

We evaluate the performance of EBM (the proposed metric) using forward ensemble selection: starting with an initial vector $\mathbf{w}$ to be $\mathbf{0}$ and iteratively setting one item w_i in $\mathbf{w}$ to be 1, where $\mathbf{w}$ is the weight vector associated with ensemble members. Complementariness (COM) [10], concurrency (CON) [11] and uncertainty weighted accuracy (UWA) [16] are used as the compared metrics.
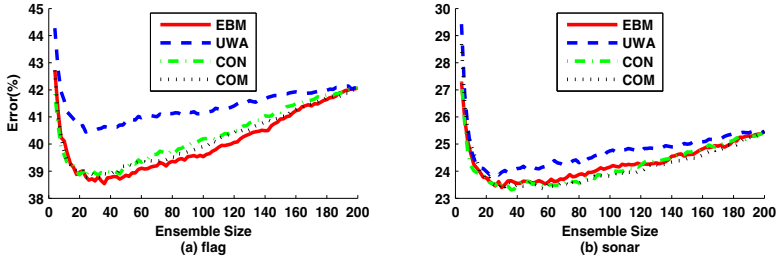
Fig. 3. Comparative results for two data sets in the second category
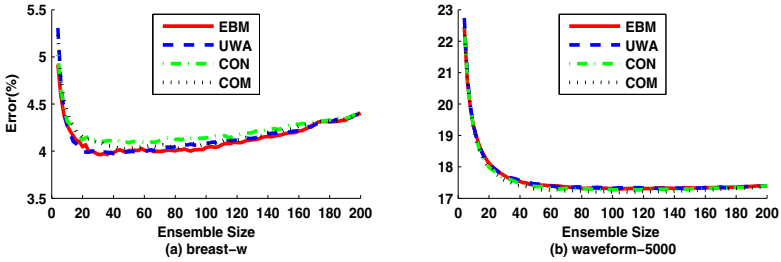
Fig. 4. Comparative results for two data sets in the third category
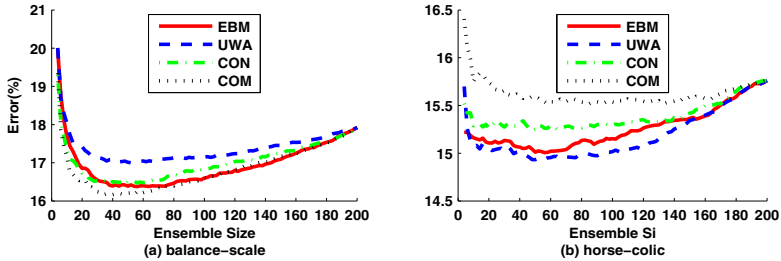
Fig. 5. Comparative results for two data sets in the forth category

In each trial, a bagging is trained. The base classifier is J48, which is a Java implementation of C4.5 [18] from Weka [19]. In all experiments, the ensemble size is 200.

5.2   Experimental Results

The experimental results of the 24 tested data sets are reported both in figures and a table. The figures show the curves of average error with regard to the size of subensembles, where the results of these data sets are grouped into four categories: (1) EBM outperforms UWA, CON and COM, (2) EBM performs comparable to at least one of them, and outperforms the others, (3) the four

metrics perform similar to each other, and (4) EBM is outperformed by at least one of them. 17 data sets fall into the first category, 2 data sets fall into the second category, 2 data sets fall into the third category and 3 data sets fall into the last category. The table is the comparison summary between EBM and other metrics on all data sets. The more details about this table will be introduced when describing the table.

Fig. 2 reports the error curves of the four compared metrics for six representative data sets that fall into the first category. The corresponding standard deviations are not shown in the figures for clarity reason. Fig. 2(a) is the error curves for "audiology". As shown in Fig. 2(a), with the increase of the number of aggregated classifiers, the error curve of subensembles selected by these metrics drops rapidly, reaches the minimum error in the intermediate steps of aggregation which is lower than the error of the whole original ensemble, and then increases until the error is the same as the whole ensemble. Fig. 2 also shows that the error curve of EBM is always below the error curve of UWA, CON and COM. The remainder five data sets "autos", "car", "ecoli", "soybean" and "zoo" have similar results to "audiology".

Fig. 3 reports the error results for the two data sets in the second category, where EBM performs similar to at least one of UWA, CON and COM, and outperforms others. The error curves for "flag" and "sonar" are reported in Fig. 3(a) and Fig. 3(b) respectively. As shown in Fig. 3, EBM outperforms UWA, and performs comparable to the others.

Fig. 4 reports the error results for the two data sets in the third category, where the four metrics perform similar to each other. An interesting observation from Fig. 4(b) is that, the four error curves drops to minimum error until the ensemble size is 200. This fact indicates that ensemble selection is not suitable for some data set.

Fig. 5 reports the error curves for two representative data sets in the forth group, where EBM is outperformed by at least one of UWA, CON and COM. Fig. 5(a) and Fig. 5(b) show that, although EBM is outperformed by one of other metrics on "balance-scale" and "horse-colic" respectively, the goodness is not significant. In addition, EBM outperform the other metrics, especially on "horse-colic".

Table 1 summarizes the mean and standard deviation of 300 trials for each data set. For reference, the accuracy of bagging (the whole ensembles) is displayed as well. Experimental results in this paper empirically show that these selection methods generally reach minimum errors with subensemble size to be between 20 and 40 (10%-20% of the original ensemble size). Therefore, the subensembles formed by EBM with 30 original ensemble members are compared with subensembles formed by UWA, CON and COM with the same size. A bullet next to a result indicates that EBM is significantly better than the respective method (column) for the respective data set (row). An open circle next to a result indicates that EBM is significantly worse than the respective method.

Table 1. The mean accuracy and standard deviation of EBM, UWA, CON, COM and bagging

Dataset	EBM	UWA	CON	COM	bagging
anneal	1.73±0.95	1.92±1.03 ●	2.01±0.97 ●	2.12±1.00 ●	2.82±1.15 ●
audiology	23.86±3.92	25.08±4.18 ●	24.60±4.22 ●	24.93±4.08 ●	27.37±4.72 ●
autos	28.15±5.54	29.99±5.58 ●	30.31±5.54 ●	30.34±5.41 ●	35.00±6.23 ●
balance-scale	16.55±2.25	17.14±2.27 ●	16.53±2.30	16.24±2.21 ○	17.92±2.45 ●
breast-w	3.96±1.16	3.98±1.19	4.12±1.18 ●	4.09±1.16 ●	4.40±1.33 ●
car	11.00±1.42	11.66±1.44 ●	11.49±1.48 ●	11.51±1.50 ●	13.01±1.54 ●
ecoli	16.37±3.33	16.79±3.36 ●	16.66±3.27 ●	16.74±3.18 ●	18.30±3.45 ●
flag	38.64±5.79	40.51±5.82 ●	38.93±5.92	38.82±5.90	42.06±6.22 ●
glass	29.93±5.27	29.58±5.58 ●	29.27±5.24 ●	29.61±5.14 ●	32.31±6.05 ●
heart-statlog	18.69±3.73	18.73±3.71	18.70±3.62	18.97±3.58	20.23±4.25 ●
horse-colic	15.13±2.74	15.03±2.75	15.27±2.76	15.66±2.84 ●	15.76±3.04 ●
hypothyroid	0.56±0.22	0.57±0.23 ●	0.57±0.23 ●	0.59±0.26 ●	0.69±0.30 ●
kr-vs-kp	0.85±0.31	0.88±0.32 ●	0.94±0.36 ●	1.05±0.43 ●	1.34±0.52 ●
mofn-3-7-10	12.69±3.18	13.30±2.92 ●	12.77±2.94	12.00±3.06 ○	14.44±3.22 ●
page-blocks	2.84±0.35	2.86±0.37 ●	2.88±0.36 ●	2.90±0.37 ●	3.04±0.40 ●
primary-tumor	58.67±3.50	59.27±3.46 ●	58.80±3.51	59.14±3.64 ●	60.23±3.75 ●
promoters	15.22±7.85	15.54±7.85	16.06±7.53 ●	15.82±6.59	21.54±7.13 ●
segment	3.92±0.84	4.09±0.84 ●	4.10±0.86 ●	4.04±0.82 ●	4.70±0.89 ●
sick	1.51±0.36	1.53±0.37 ●	1.56±0.36 ●	1.58±0.36 ●	1.78±0.39 ●
sonar	23.40±4.87	23.96±4.99 ●	23.53±5.10	23.55±4.97	24.47±5.20 ●
soybean	9.23±2.00	9.85±2.02 ●	9.73±2.11 ●	9.78±2.04 ●	12.05±2.46 ●
waveform-5000	17.72±0.89	17.73±0.90	17.67±0.87	17.57±0.89 ○	17.40±0.91 ○
wine	6.48±4.47	7.11±4.67 ●	7.55±4.51 ●	6.79±3.93	10.24±4.71 ●
zoo	8.79±4.34	9.34±4.39 ●	9.45±4.28 ●	9.58±4.66 ●	10.77±4.84 ●
win/tie/loss		19/5/0	16/8/0	16/5/3	23/0/1

● *represents that EBM outperforms the comparing method in pair-wise t-tests at 95% significance level, and* ○ *suggests that EBM is outperformed by comparing method.*

As shown in table 1, EBM outperforms bagging on 23 out of 24 data sets, which indicates that EBM efficiently performs selection by achieving better predictive accuracies with small subensembles. In addition, EBM is shown to significantly outperform UWA, CON and COM in 19, 16 and 16 out of 24 data sets respectively. This fact indicates that EBM is a useful metric to supervise greedy ensemble selection algorithms.

Combining the results all above, we conclude that EBM is a rather ideal metric to supervise greedy ensemble selection algorithms.

6   Conclusion

This paper contributes a novel viewpoint (energy-based learning) for ensemble problem, based on which a new metric called EBM (Energy-based Metric) is proposed to supervise greedy ensemble selection. Also, this metric considers the strength of the decision of current ensemble.

Our experiments show that, compared with state-of-the-art ensemble selection methods, EBM based method has significantly better generalization capability in most of data sets. This result indicates that energy-based learning is valuable for ensemble selection. In addition, we consider that the new viewpoint (energy-based learning) for ensemble problem is positive contribution that is valuable to other researchers working in ensemble problems in general.

References

1. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. John Wiley and Sons (2004)
2. Breiman, L.: Bagging predictors. Machine Learning, 123–140 (1996)
3. Freund, Y., Schapire, R.F.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139 (1997)
4. Breiman, L.: Random forests. Machine learning 45, 5–32 (2001)
5. Zhang, D., Chen, S., Zhou, Z., Yang, Q.: Constraint projections for ensemble learning. In: 23rd AAAI Conference on Artificial Intelligence (AAAI 2008), pp. 758–763 (2008)
6. Zhang, Y., Burer, S., Street, W.N.: Ensemble pruning via semi-definite programming. The Journal of Machine Learning Research, 1315–1338 (2006)
7. Chen, H., Tino, P., Yao, X.: Predictive ensemble pruning by expectation propagation. IEEE Transactions on Knowledge and Data Engineering, 999–1013 (2009)
8. Guo, H., Fan, M.: Ensemble Pruning via Base-Classifier Replacement. In: Wang, H., Li, S., Oyama, S., Hu, X., Qian, T. (eds.) WAIM 2011. LNCS, vol. 6897, pp. 505–516. Springer, Heidelberg (2011)
9. Tamon, C., Xiang, J.: On the Boosting Pruning Problem. In: Lopez de Mantaras, R., Plaza, E. (eds.) ECML 2000. LNCS (LNAI), vol. 1810, pp. 404–412. Springer, Heidelberg (2000)
10. Martinez-Muverbnoz, G., Suarez, A.: Aggregation ordering in bagging. In: International Conference on Artificial Intelligence and Applications (IASTED), pp. 258–263. Acta Press, Calgary (2004)
11. Banfield, R.E., Hall, L.O., Bowyer, K.W., Kegelmeyer, W.P.: Ensemble diversity measures and their application to thinning. Information Fusion, 49–62 (2005)
12. Partalas, I., Tsoumakas, G., Vlahavas, I.P.: Focused Ensemble Selection: A Diversity-Based Method for Greedy Ensemble Selection. In: 18th European Conference on Artificial Intelligence, pp. 117–121 (2008)
13. Martínez-Muñoz, G., Hernández-Lobato, D., Suárez, A.: An analysis of ensemble pruning techniques based on ordered aggregation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 245–259 (2009)
14. LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M.A., Huang, F.J.: A Tutorial on Energy-Based Learning. In: Predicting Structured Data. MIT Press (2006)
15. Fan, W., Chu, F., Wang, H., Yu, P.S.: Pruning and dynamic scheduling of cost-sensitive ensembles. In: Eighteenth National Conference on Artificial Intelligence. American Association for Artificial Intelligence, pp. 146–151 (2002)
16. Partalas, I., Tsoumakas, G., Vlahavas, I.P.: An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. Machine Learning, 257–282 (2010)
17. Asuncion, D.N.A.: UCI machine learning repository (2007)
18. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann (1993)
19. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann (2005)

PUF-Based RFID Authentication Protocol against Secret Key Leakage

Yongming Jin[1,2], Wei Xin[1,2], Huiping Sun[3], and Zhong Chen[1,2]

[1] School of Electronics Engineering and Computer Science
Peking University, Beijing, China
[2] Key Laboratory of High Confidence Software Technologies
Ministry of Education, Beijing, China
[3] School of Software and Microelectronics
Peking University, Beijing, China
{jinym,xinwei,chen}@infosec.pku.edu.cn, sunhp@ss.pku.edu.cn

Abstract. RFID tags are now pervasive in our everyday life. They raise a lot of security and privacy issues. Many authentication protocols against these problems assume that the tags can contain a secret key that is unknown to the adversary. However, physical attacks can lead to key exposure and full security breaks. On the other hand, many protocols are only described and analyzed. However, we cannot explain why they are designed like that. Compare with the previous protocols, we first propose a universal RFID authentication protocol and show the principle why the protocol is designed. It can be instantiated for various types and achieve different security properties according to the implementation of the functions. Then we introduce a general prototype of delay-based PUF for low-cost RFID systems and propose a new lightweight RFID authentication protocol based on the general prototype of PUF. The new protocol not only resists the physical attacks and secret key leakage, but also prevents the asynchronization between the reader and the tag. It also can resist the replay attack, man-in-the-middle attack etc. Finally, we show that it is efficient and practical for low-cost RFID systems.

Keywords: Protocol, Security, RFID, Authentication, Physical Unclonable Function.

1 Introduction

Radio Frequency Identification (RFID) is an automatic identification method based on reading and writing data remotely at certain frequency between devices. RFID tags are small, inexpensive devices that communicate wirelessly with RFID readers. Most RFID tags currently in use are passively powered and respond to queries from legitimate, but also rogue RFID readers. Recently, the wide deployment of RFID systems in a variety of applications has raised many concerns about the security and privacy.

The security of classical authentication protocols usually relies on a computationally hard problem that most constructions are based on a secret key which is assumed to possess in the hands of the entity of protocols. However, it is difficult to satisfy this requirement in practical scene. There are many physical attacks that can lead to key exposure and break the full security. For example, the side-channel attacks[1], memory invasion[2], etc. Classical cryptography does not provide a secure way to prevent these attacks.

Physical Unclonable Function (PUF), was proposed to as a building block for authentication schemes that resist physical attacks. PUF is first introduced by Pappu[3] as a disordered physical system that can receive external challenges and return the corresponding responses. A PUT's responses depend on the structural disorder present in the PUT. This disorder cannot be cloned or reproduced exactly, not even by its original manufacturer, and is unique to each PUF[4].

Because the PUF is complex and has disordered structure, it is usually harder to read out, predict, or derive its responses than to obtain the values of digital keys for various memory. So it can solve some of the shortcomings associated with digital keys. There are many PUF-based security protocols for identification, authentication, or key exchange purposes [5-8].

The remainder of the paper is organized as follows. In Section 2, related work is reviewed. In Section 3, we propose a universal RFID authentication protocol. We introduce a prototype of the implementation of physical unclonable function in Section 4. We propose the PUF-based RFID authentication protocol against secret key leakage in Section 5. In Section 6, the security and privacy is analyzed. In Section 7, we compare the efficiency of the relative protocols. Finally, we conclude our paper in Section 8.

2 Related Work

Many RFID authentication protocols are proposed. We focus on the problem of secret key leakage and review some related work. We also introduce the latest developments of PUFs and the research in the RFID authentication protocols.

An RFID privacy protection scheme is proposed by Ohkubo et al. providing indistinguishability and forward security[9]. To achieve forward security, they use the hash chain technique to renew the secret information contained in the tag. However, it is subject to replay attacks, and it permits an adversary to impersonate a tag without knowing the tag secrets.

Molnar and Wagner focus on the privacy issues related to RFID technology in libraries and propose a general scheme for building private authentication with work logarithmic in the number of tags[10]. However, it is not forward security. Once a tag is compromised, the attacker can trace past communications from this tag, because a tag's identifier and secret key are static.

A simple authentication algorithm for RFID system based on PUF is proposed by Ranasinghe et al.[11] In this scheme, the database learns many challenge-response pairs for each tag's PUF circuit, and uses hundreds of challenges at a time to identify and authenticate millions of tags, probabilistically ensuring unique identifications. However,

there are some disadvantages in this scheme. The lack of access control provisions exposes tags to identification by attack. The tags do not maintain a state and do not use any randomness in their responses, making them vulnerable to tracking.

Dimitriou proposes an RFID authentication protocol that enforces user privacy and protects against tag cloning[12]. The protocol is based on the use of a secret shared between tag and database that is refreshed to avoid tag tracing. However, the scheme is prone to the asynchronization attack.

The LMAP protocol, a lightweight mutual authentication protocol for low-cost RFID tags, is proposed by Lopez et al.[13] It is efficient and can be implemented in the most RFID systems that only need around 300 gates. Lopez et al. also proposed a M^2AP protocol that has the similar properties[14]. However, the attacker can break the synchronization between the RFID reader and the tag in a single protocol run so that they cannot authenticate each other in any following protocol runs[15].

Tuyls and Batina investigate how an RFID tag can be made unclonable by linking it inseparably to a PUF unit. Then an off-line reader authentication algorithm based on PUFs is proposed. The algorithm uses public key cryptography, which is still prohibitively expensive for low-cost RFID tags[16].

Based on the EPC Class 1 GEN-2 standards, a mutual authentication protocol for RFID is proposed by Chien and Chen[17]. The database maintains copies of both old and new tag keys to resist the asynchronization attack. In order to give forward security, the authentication key and the access key are updated after a successful session. However, a strong attacker that compromises a tag can identify a tag's past interactions from the previous communications.

Bolotnyy and Robins propose hardware-based approaches to RFID security that rely on physical unclonable functions[18]. They describe protocols for privacy-preserving tag identification and secure message authentication codes and compare PUFs to digital cryptographic functions. The proposed solutions are efficient, practical, and appropriate for low-cost RFID systems.

A lightweight challenge response authentication scheme based on noisy physical unclonable function is proposed by Ozturk et al. [6] The proposed authentication scheme may be implemented in a very small footprint and deployed in applications with stringent power limitations. Furthermore, the inherent properties of PUFs provide cryptographically strong tamper resilience–a feature crucial for applications where the device is in the hand of potential attackers.

Berbain et al. proposed a novel forward private authentication scheme build upon less computationally expensive cryptographic ingredients instead of one way hash functions[19]. The new protocol is based on less complex cryptographic building blocks. This yields efficient hardware implementations compared with previous RFID protocols.

Ma et al. refine the definition of unp-privacy and proven that ind-privacy is weaker than unp-privacy. In this sense, a pseudorandom function family is the minimal requirement on an RFID tag's computational power for enforcing strong RFID system privacy. They also propose a new RFID protocol that satisfy the minimal requirement[20].

Cortese et al. propose new methodologies for the authentication of RFID tags along supply chains, exploiting tags equipped with a PUF device[7]. It achieves a constant amount of shared secret data that can be distributed using a secure hardware

token. The rest of the data can be released over an insecure one-way communication channel that can be realized by shipping storage media along with goods.

Choi et al. propose a PUF-based encryption processor and the low-cost RFID authentication protocol that the challenge-response pairs are encrypted by the PUF's characteristics[21]. The PUF-based encryption processor consists of N-bit PUF, encryption, decryption, ECC modules. It is implemented at low cost with small footprint and low power. The proposed scheme defends several kinds of attacks, which are physical, modeling, spoofing and location tracking attacks.

A lightweight RFID mutual authentication and ownership transfer protocols is proposed by Kulseng et al.[8] Both protocols use a combination of PUF and Linear Feedback Shift Register (LFSR) that are very efficient in hardware and particularly suitable for the low-cost RFID tags. However, there are several serious security issues with these protocols[22]. The mutual authentication protocol suffers from block messaging attack which breaks the synchronization between the RFID reader and the tag. The protocols are not resistant message injection attack because of the lack of message integrity.

3   Universal RFID Authentication Protocol

In this section, we propose a universal RFID authentication protocol. We use the definitions as described in Section 5.

In the initialization phase, the tag manufacturer assigns a identifier *ID* and chooses a secret key k. They are stored in the tag and the reader. The reader also stores the old secret key k' in order to resist the asynchronization attack. At the beginning, $k'=k$.

In order to resist the replay attack, the reader and the tag should produce random numbers in every authentication session. Otherwise, the attacker may use the old messages in another authentication process because these messages keep static. In a sense, a protocol may be susceptible to the replay attack if the protocol does not produce a random number or other similar measures. In the first step of our universal RFID authentication protocol, the reader R_i produces a random number $r \in \{0,1\}^l$ and sends r to the tag. The tag T_i also produces a random number $r' \in \{0,1\}^l$. In some RFID authentication protocols, the reader only sends request query and doesn't produce a random number. It seems that it is enough if the tag can produce a random number. However, if the reader does not keep its token r, the tag may compute M_1 and M_2 according to his wishes. The attacker also can run the replay attack towards the reader.

In the next step, the tag need send a Message Authentication Code (MAC) to the reader in order to authenticate itself. We define a function h that produces the MAC M_2. In general, h should be a one-way function and secure enough. We also define a function f that produces the message M_1. f is a reversible function that the reader can compute the tag's random number r'.

When the reader receives the messages M_1 and M_2, it searches for the tuple (ID,k) such that $M_2=h(r,r',ID)$ where $r'=f^{-1}(k,M_1)$. If such a tuple exists, the reader continues the next step. Otherwise, it repeats the search with the tuple (ID,k'). If no match is found, the reader stops the session. Otherwise, the reader has identified and

authenticated T_i. Then it computes the message $N=g(k,ID,r')$ and sends N to the tag. The reader computes $k_{new}=p(k,ID,r')$ and updates $k=k_{new}$, $k'=k$. In the easier way, the function g can be a HASH function or other MACs. However, when the tag checks the N, it will increase the tag's cost of computation. Therefore, the best way is to design the function g in a lightweight method.

Because the channel between the reader and the tag is unsecure and the attacker can block or modify the messages at any time, the tag may lose the message N and does not update k. However, the reader has done. In another authentication process, the reader will receive the wrong messages M_1 and M_2 though the tag is valid. To solve this problem, we store the old secret key k' in the reader. The authentication process can be restored by searching the tuple (ID,k').

In order to resist the attacker's tracking, the tag can not send a message as a index to the reader. Therefore, the reader has to search the records linear in the number of tags. It is possible that the tag sends more MAC messages that contains the index. However, it will increase the tag's cost clearly. Though some protocols propose some measures, they are not perfect. For example, the Ma et al. protocol uses I as a index in the exact match. In order to protect the Unp-Privacy, I is a PRF function. So the tag needs compute two MAC messages and send them to the reader. However, we only need compute one MAC message. Additionally, this protocol cannot resist the DoS attack. On the other hand, we think that the capability of the reader is enough to satisfy this requirement under the assumptions that an RFID reader is usually equipped with enough computational power. We assume that the reader is not resource-limited and focus on the minimal requirement for RFID tags only the tag is lower cost.
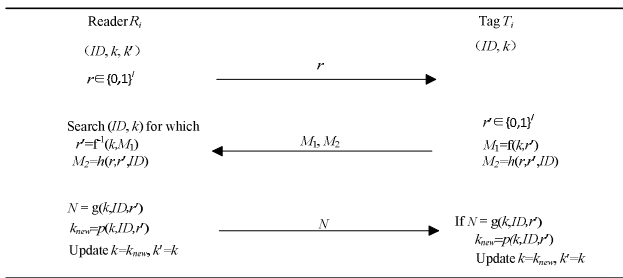

Fig. 1. Universal RFID Authentication Protocol

4   Physical Unclonable Function

The silicon PUF utilizes uniquely physical characteristics on each IC. Owing to the manufacturing process, each IC has different delays for the gate and the wire. Therefore, the PUF on each IC has different output in spite of the same logic design. In general, a PUF satisfies the following properties [23]: (1) It is impossible to build another PUF that has the same response behavior. (2) It is hard to predict the output of a PUF for a given input without performing and measurement. (3) The output looks random.

First, we introduce a prototype of delay-based PUF as shown in Figure 2. The PUF is a P: $\{0, 1\}^k \rightarrow \{0, 1\}$ mapping, that takes a k-bit challenge a and produces a single output bit r. It consists of switching boxes and an arbiter. The arbiter-based PUF is suitable for limited-resource platforms like the low-cost RFID tag. The distribution of the delay values for different challenges tends to be Gaussian, with many challenges producing identical (or similar) outputs even when signals take different paths through the delay circuit.
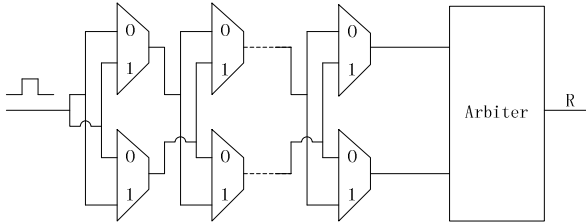

Fig. 2. The prototype of delay-based PUF

However, in the linear delay model[24], the attacker can express the final delay difference between the upper and the lower path in a delay-based PUF. The overall delays of the signals are modeled as the sum of the delays in the states. In order to resist this attack, we can increase the number of the Arbiter. For example, a standard fabrication processes implement an XOR Arbiter PUF with 8 XORs and bit length of 512. It is secure enough[25].

On the other hand, a better PUF circuit could leverage sub-threshold voltage techniques to compare gate polarizations, thus running quickly without using an oscillating counter[5]. Such methods can expect to better separate PUF values for different challenges, and thus avoid highly skewed distributions of PUF responses while still preserving PUF reliability and unpredictability. To keep the PUF modeling difficult, variable non-linear delays can add to the circuit.

In order to raise the attacker's complexity, we propose an integrated scheme of PUF for low cost RFID as shown in Figure 3. Compare with the previous schemes, we introduce the nonlinearity elements in the new scheme, such as AND, OR gates. In some cases, an attacker may simulate the original PUF unit and predict the responses of random challenges. We use the forward feedback arbiter to resist this attack. For the internal bits of forward feedback are hidden to the attacker, it is very hard to build a precise model of the forward feedback arbiter PUF.
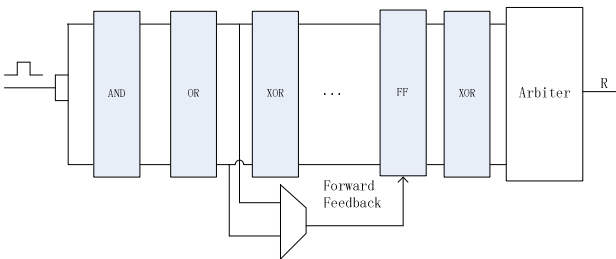

Fig. 3. An Integrated Scheme of PUF for low cost RFID tags

5   Proposed Protocol

5.1   Definitions

We use the following definitions.

T_i: an RFID tag
R_i: an RFID reader
l: the length of secret string that is a even number
G_i: the secret key of T_i
U_i: the detailed information associated with tag T_i
F: the Hash function, $\{0,1\}^l \rightarrow \{0,1\}^{l'}$
$\oplus$:   XOR operator
$\leftarrow$:   Substitution operator
P: the PUF function
$x >> a$: the right circular shift operator, which rotates all bits of x to the right by the bits, as if the right and left ends of x are joined.

5.2   Protocol Description

In this protocol, we use G_{n-1} as the secret key k, and store the old secret key G_n in the reader. In order to resist the secret key leakage, the tag first computes $G_n=P(G_{n-1})$ and continues the authentication process with G_n. In some sense, we can view G_n as the secret key. We design the function $f(k,r)= G_n \oplus G_{n+1}$, $h(r,r',ID)= F(ID \oplus G_{n+1} \oplus r)$. The parameter ID has two functions. First, it is the tag's identifier. The reader can identify one tag and check if it is valid. Second, ID is secret to the adversary in order to resist the tracking. Therefore, we design $N=ID \oplus (G_{n+1} >> l\ /2)$ to authenticate the reader. Because the ID and G_{n+1} are secret, the attacker can not obtain any information. The right circular shift operator can void the combination M_1 and N. The attacker may forge another authenticated messages to cheat the tag. We describe the protocol as follow.

1)   **$R_i \rightarrow T_i$: r**
The reader R_i generates a random bit-string r and sends it to T_i.

2)   **$T_i \rightarrow R_i$: M_1, M_2.**
The tag computes $G_n=P(G_{n-1})$, $G_{n+1}=P(G_n)$, $M_1=G_n \oplus G_{n+1}$, $M_2=F(ID \oplus G_{n+1} \oplus r)$, and sends ($M_1$, M_2) to the reader.

3)   **$R_i \rightarrow T_i$: N**
For each tuple (ID,G_n), R_i computes $G_{n+1} = M_1 \oplus G_n$ and verifies whether the equations $M_2=F(ID \oplus G_{n+1} \oplus r)$. If it can find a match, then the tag T_i is successfully identified and authenticated. If it can't find a match, R_i computes $G_{n+1} = M_1 \oplus G_{n-1}$ and verifies whether the equations $M_2=F(ID \oplus G_{n+1} \oplus r)$ hold for each tuple (ID,G_{n-1}). It is means that the tag does not update its key from G_{n-2} to G_{n-1} in the last authentication

process. The tag computed M_1 by G_{n-1} and G_n. If there is a match, the reader computes $N=ID\oplus(G_{n+1}>>l/2)$, and sends N to T_i. Finally, R_i updates the old value (G_{n-1},G_n) by (G_n,G_{n+1}) respectively.

4)   The Tag T_i

T_i computes $N'=ID\oplus(G_{n+1}>>l/2)$. If $N=N'$, then update G_{n-1} by G_n.
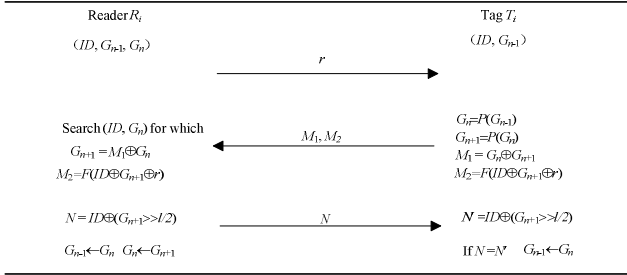

Fig. 4. PUF-based Lightweight RFID Authentication Protocol

6   Security and Privacy

In this section, we give a security and privacy analysis of our proposed scheme. The most important point is that the protocol forms a chain of secret key G_i when the reader and the tag are mutually authenticated. If G_{n+1} is attacked and wrong, the tag can discover this attack and does not update its secret key. Then it can be restored in next authentication process.

Key Leakage (KL)

Physical attacks, such as side-channel attack, memory invasion can lead to key exposure. The new protocol can resist the key leakage and provide backward security. Based on the PUF function P, the protocol builds a secret keys' chain. In each authentication process, G_n is the secret key. However, we only store the previous secret key G_{n-1}. Even if the current tag secret G_{n-1} has been revealed, the attacker can not forge the tag and get the same PUF function that has same response. Therefore it can not compute the next secret G_n and continue the future transactions.

Replay Attack (RA)

An adversary may try to impersonate an authenticated tag or reader by the old messages. For example, we denote the previous round messages $M_1'=G_{n-1}\oplus G_n$, $M_2'=F(ID\oplus G_n\oplus r')$. If the attacker replaces (M_1, M_2) with (M_1', M_2'), the reader computes $M_2=F(ID\oplus G_{n+1}\oplus r)=F(ID\oplus M_1'\oplus G_n\oplus r)=F(ID\oplus G_{n-1}\oplus G_n \oplus G_n\oplus r)$ $=F(ID\oplus G_{n-1}\oplus r)$. In order to authenticate the tag, M_2 should be equal to M_2'. Because r is random that produced by the reader and G_n is secret and unclonable, it is hard to succeed. Even if the attacker find a way to make $M_2=M_2'$, the third message N is relative to G_{n+1}. If the tag find N is not equal to N', G_{n-1} will not be updated. Then the reader and the tag can mutually authenticated by G_n in next round.

Denial of Service (DoS)

To resist the *DoS* attack, we require the reader to save the old values to recover synchronization with T_i. If an adversary prevents message N from reaching T_i, T_i will not update its identifier, but R_i will. In the following authentication process, T_i will use old secret value to compute M_1, M_2 and R_i can recover the old secret value from (ID, G_{n-1}).

Man-in-the-middle Attack (MITM)

Any attacker can obtain the message (M_1, M_2) and N. He can prevent the correct message and send a different message. If he changes the old message, he can modify r, M_1, M_2, and N. If he modifies N, the reader will not be authenticated that deduce to the asynchronization with T_i. In the next authentication process, the reader can be authenticated by old secret key. If he modifies M_2, he must resolve the hard problem of Hash function. It is very hard. We suppose the attacker replace r with $r \oplus x$, M_1 with $M_1 \oplus y$. The reader receives $M_1' = G_n \oplus G_{n+1} \oplus y$, $M_2' = F(ID \oplus G_{n+1} \oplus r \oplus x)$. The reader computes $M_2 = F(ID \oplus G_{n+1} \oplus r) = F(ID \oplus M_2' \oplus G_n \oplus r)$ $= F(ID \oplus G_n \oplus G_{n+1} \oplus y \oplus G_n \oplus r) = F(ID \oplus G_{n+1} \oplus r \oplus y)$. If M_2 is equal M_2', x is equal to y. That means the attacker must modify the r and M_1 with the same string. If the attacker let the reader get the wrong G_{n+1}, N will not be correct. The authentication process can be recovered.

The following table shows the comparison in the sense of security discussed in this Section.

Table 1. Security and Privacy Comparison

Schemes	KL	RA	DoS	MITM
Ohkubo et al.[9]	√	×	√	√
Molnar et al.[10]	*	√	√	√
Dimitriou[12]	√	√	×	√
Lopez et al.[13]	√	√	×	√
Chien et al.[17]	√	√	√	√
Berbain et al.[19]	√	×	×	√
Ma et al.[20]	√	√	×	√
Kulseng et al.[8]	√	√	×	×
The new scheme	√	√	√	√

√: resist such an attack.

×: can't protect against such an attack.

*: resists attack under some assumptions

7   Performance Comparison

In the new protocol, the tag needs k bits of non-volatile memory to store its secret G_{n-1}. It is common condition that we can find out through the following table. The tag also stores its identity *ID* to identify itself. Compare with other schemes, the new protocol has lower tag computation and tag communication cost.

Obviously, the new protocol needs the reader more storage cost to storage the old secret value (G_{n-1}, G_n) to prevent the asynchronization between the reader and the tag when the tag fails to receive the last message authentication code. Let k is the length of the random number and the secret value of T_i, l is the length of the tag's *ID*, q is the length of hash value. Let the number of total tags is n, p is the cost of PRNG operation, u is the cost of PUF function, h is the cost of *Hash* function, r is the cost of LFSR. In general, r is less than h. The cost of XOR, bit shift, etc. operation is negligible.

The table 2 shows the performance comparison between the existing scheme and the new scheme.

Table 2. Performance Comparison

Schemes	TC	TS	RC	RS	CC
Ohkubo et al.[9]	$2h$	k	$2h$	$l+k$	l
Molnar et al.[10]	$h+p$	$l+k$	$O(n)h+p$	$l+k$	$2l+2k$
Dimitriou[12]	$2h+p$	l	$2h+p$	k	$2k+3q$
Lopez et al.[13]	$O(1)$	$l+k$	$2p$	$l+k$	$5k$
Chien et al.[17]	p	$l+k$	p	$l+k$	$2k+2q$
Berbain et al.[19]	$3h$	k	$O(n)h+p$	$l+k$	$k+q$
Ma et al.[20]	$2h$	$k+q$	$2h+p$	$2q+l+k$	$k+2q$
Kulseng et al.[8]	$3r+2u$	$2l+k$	$r+p'$	$2k+2l$	$2k+3l$
The new scheme	$h+2u$	$k+l$	$O(n)h$	$l+2k$	$k+l+q$

TC: Tag Computation cost.
TS: Tag Storage cost.
RC: Reader Computation cost.
RS: Reader Storage cost.
CC: Communication cost.

8   Conclusions

Instead of attacking the cryptographic algorithms used in RFID systems, the adversary often chooses to attack the hardware in which a secret key is stored, such as invasive, semi-invasive, or side-channel attacks. In this paper, we propose a novel lightweight RFID authentication protocol based on PUF. The new protocol can avoid the shortcomings associated with secret key leakage because the PUF's response depend on a nanoscale structural disorder that cannot be cloned or reproduced exactly, not even by its original manufacturer, and is unique to each PUF. Compare with the previous PUF-based protocols, it is lightweight and provides the mutual authentication between the reader and the tag. It can also resist the asynchronization attack as same as replay attack, man-in-the-middle attack etc. We also show that it requires less cost of computation and storage than other similar protocols.

In the future work, it would be interesting to develop the PUF-based RFID authentication protocol that has the property of forward security and bring the protocol into the FPGA implementation. The new protocol may be developed for

different applications, including multi-tag regimes [26]. It can also benefit RFID ownership transfer scene [8, 27, 28].

References

1. Kasper, T., Oswald, D., Paar, C.: Side-Channel Analysis of Cryptographic RFIDs with Analog Demodulation. In: Juels, A., Paar, C. (eds.) RFIDSec 2011. LNCS, vol. 7055, pp. 61–77. Springer, Heidelberg (2012)
2. de Koning Gans, G., Hoepman, J.-H., Garcia, F.D.: A Practical Attack on the MIFARE Classic. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 267–282. Springer, Heidelberg (2008)
3. Pappu, R.: Physical one-way functions. Massachusetts Institute of Technology (2001)
4. Rührmair, U., et al.: Modeling attacks on physical unclonable functions. In: Proceedings of the 17th ACM Conference on Computer and Communications Security. ACM, Chicago (2010)
5. Gassend, B., Clarke, D., Van Dijk, M., Devadas, S.: Silicon physical random functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. ACM (2002)
6. Oztiirk, E., Hammouri, G., Sunar, B.: Towards robust low cost authentication for pervasive devices. In: Sixth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2008. IEEE (2008)
7. Cortese, P.F., Gemmiti, F., Palazzi, B., Pizzonia, M., Rimondini, M.: Efficient and practical authentication of PUF-based RFID tags in supply chains. In: 2010 IEEE International Conference on RFID-Technology and Applications (RFID-TA). IEEE (2010)
8. Kulseng, L., Yu, Z., Wei, Y., Guan, Y.: Lightweight mutual authentication and ownership transfer for RFID systems. In: Proceedings IEEE INFOCOM. IEEE (2010)
9. Ohkubo, M., Suzuki, K., Kinoshita, S.: Cryptographic approach to "privacy-friendly" tags. In: RFID Privacy Workshop (2003)
10. Molnar, D., Wagner, D.: Privacy and security in library RFID: Issues, practices, and architectures. In: Proceedings of the 11th ACM Conference on Computer and Communications Security. ACM (2004)
11. Ranasinghe, D., Engels, D., Cole, P.: Security and privacy: Modest proposals for low-cost RFID systems. In: Proceedings of the Intelligent Sensors, Sensor Networks and Information Processing Conference. IEEE (2004)
12. Dimitriou, T.: A lightweight RFID protocol to protect against traceability and cloning attacks. In: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks 2005. IEEE Computer Society (2005)
13. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A.: LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags. In: Proceedings of 2nd Workshop on RFID Security (2006)
14. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A.: M^2AP: A Minimalist Mutual-Authentication Protocol for Low-Cost RFID Tags. In: Ma, J., Jin, H., Yang, L.T., Tsai, J.J.-P. (eds.) UIC 2006. LNCS, vol. 4159, pp. 912–923. Springer, Heidelberg (2006)

15. Li, T., Wang, G.: Security analysis of two ultra-lightweight RFID authentication protocols. In: New Approaches for Security, Privacy and Trust in Complex Environments, pp. 109–120 (2007)
16. Tuyls, P., Batina, L.: RFID-Tags for Anti-counterfeiting. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 115–131. Springer, Heidelberg (2006)
17. Chien, H.-Y., Chen, C.-H.: Mutual authentication protocol for RFID conforming to EPC Class 1 Generation 2 standards. Computer Standards & Interfaces 29(2), 254–259 (2007)
18. Bolotnyy, L., Robins, G.: Physically unclonable function-based security and privacy in RFID systems. In: Fifth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2007. IEEE (2007)
19. Berbain, C., Billet, O., Etrog, J., Gilbert, H.: An efficient forward private RFID protocol. In: Proceedings of the 16th ACM Conference on Computer and Communications Security. ACM (2009)
20. Ma, C., Li, Y., Deng, R.H., Li, T.: RFID privacy: relation between two notions, minimal condition, and efficient construction. In: Proceedings of the 16th ACM Conference on Computer and Communications Security. ACM, New York (2009)
21. Choi, W., Kim, S., Kim, Y., Park, Y., Ahn, K.: PUF-based Encryption Processor for the RFID Systems. In: 2010 IEEE 10th International Conference on Computer and Information Technology (CIT). IEEE (2010)
22. Kardas, S., Akgün, M., Kiraz, M.S., Demirci, H.: Cryptanalysis of Lightweight Mutual Authentication and Ownership Transfer for RFID Systems. In: 2011 Workshop on Lightweight Security & Privacy: Devices, Protocols and Applications (LightSec). IEEE (2011)
23. Busch, H., Katzenbeisser, S., Baecher, P.: PUF-Based Authentication Protocols – Revisited. In: Youm, H.Y., Yung, M. (eds.) WISA 2009. LNCS, vol. 5932, pp. 296–308. Springer, Heidelberg (2009)
24. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Lightweight secure pufs. IEEE Press (2008)
25. Devadas, S.: Physical Unclonable Functions and Secure Processors. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 65–65. Springer, Heidelberg (2009)
26. Bolotnyy, L., Robins, G.: Multi-tag RFID systems. International Journal of Internet Protocol Technology 2(3), 218–231 (2007)
27. Dimitriou, T.: rfidDOT: RFID delegation and ownership transfer made simple. In: Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks. ACM (2008)
28. Song, B.: RFID tag ownership transfer. In: Proceedings of RFIDSEC Workshop (2008)

Collective Viewpoint Identification of Low-Level Participation

Bin Zhao[1,2], Zhao Zhang[1], Yanhui Gu[3], Weining Qian[1,*], and Aoying Zhou[1]

[1] East China Normal University, China
{zhzhang,wnqian,ayzhou}@sei.ecnu.edu.cn
[2] Nanjing Normal University, China
zhaobin@njnu.edu.cn
[3] The University of Tokyo, Japan
guyanhui@tkl.iis.u-tokyo.ac.jp

Abstract. Mining microblogs is an important topic which can aid us to gather collective viewpoints on any event. However, user participation is low even for some hot events. Therefore, collective viewpoint discovery of low-level participation is a practical challenge. In this paper, we propose a $\underline{T}$erm-$\underline{R}$etweet-$\underline{C}$ontext (TRC) graph, which simultaneously incorporates text content and retweet context information, to model user retweeting. We first identify representative terms, which constitute collective viewpoints. And then we apply Random Walk on TRC graph to measure the relevance between terms and group them into collective viewpoints. Finally, extensive experiments conducted on real data collected from Sina microblog demonstrated that our proposal outperforms the state-of-the-art approaches.

1 Introduction

A microblog is a popular Web 2.0 system, such as Twitter and Buzz, which is now developing into a broadcast medium expressing public opinion. It allows users to post short messages, a.k.a. *tweets*, which have up to 140 characters. However tweets cover a wide variety of content, ranging from breaking news, discussion, personal life, activities and interests, etc. Towards a hot event, microblogs usually collect diverse and abundant thoughts, comments and opinions. *How do microblog users think about such an event?* This is an interesting and practical problem.

For example, consider discussions about hot events in microblogs. There is a famous microblog event "QQ vs. 360" between two companies on China Internet in 2010[1]. The event had attracted widespread attention of microblog users. They posted a large amount of tweets in a short time to express all kinds of opinions from different viewpoints. We select two examples from those tweets which are illustrated in Table 1. As the table shows, diverse viewpoints can be found from tweets.

---

[*] Corresponding author.
[1] http://www.chinadaily.com.cn/bizchina/2010-11/04/content_11501440.htm

Table 1. Sample Tweets on "QQ vs. 360" Event

	Sample Tweet	Viewpoint
1	To tackle the compatibility problem between 360 and QQ, locate the <u>folder</u> "safebase" and <u>delete</u> it, or create a new <u>text</u> document and <u>rename</u> it "safebase", finally <u>remove</u> the <u>extension</u> "txt".	Technique
2	I strongly despise <u>Tencent</u>'s and <u>Qihoo</u>'s acts harmful to <u>user interests</u>.	Criticism

In the paper, we aim to identify collective viewpoints on any given event in microblogs. However, *what are collective viewpoints like? How to represent them?* In Table 1, the first tweet shows that a user tried to solve compatibility problem between two programs. It is from the viewpoint of technique. The second one shows criticism over both companies and dislike for the whole event. It is from the viewpoint of criticism.

So many individual viewpoints like above ones will converge into collective viewpoints. In order to find them, we use representative words to depict every individual viewpoint and group them into clusters. For example, underlined words in Table 1 are representative words for presenting associated viewpoints. {folder,delete,text,rename,remove,extension} is to depict the technical viewpoint, and {despise,Tencent,Qihoo,user interests} is to depict the critical viewpoint.

The topic discovery [5,8,15,17] can exploit viewpoints from microblog datasets. They focus on document-level or sentence-level techniques for text mining. Usually they assume that only one topic is involved in one document or sentence. But this assumption is not suitable in the real world, especially for viewpoint discovery. A tweet may support multiple viewpoints due to personalized opinions. Furthermore, these techniques for long text mining are not suitable for short texts like tweets. Therefore, existing methods can not work well. In this paper, we focus on capturing term-level correlations, which drive representative terms to coverage into collective viewpoints. Another similar research work is opinion mining [4,11,13]. They focus on sentiment words such as adjectives or adverbs. However mining viewpoints is not limited to such words.

In our real applications, we note that user participation is not so active as imagined. For example, both "QQ vs. 360" event and "Expo 2010 Shanghai China"[2] were very hot in microblogs in 2010. According to our statistics, 68% users just posted one tweet during the former event, and 78% users posted one tweet during the latter event. We observe that many existing methods [1,7] usually assume that user participation is high in their case. [14] leverages users' long-term and short-term preferences for temporal recommendation. [16] models multiple user postings as time goes by. The common idea of these methods is to model precisely and effectually user behaviours. And they prefer the users with the high frequency of behaviours. However, low-level user participation often occurs during microblog events. Low participation rate may result in unavailability of multiple characteristics, such as temporal and social networking

---

[2] http://www.expo2010.cn/

information. In this way, users' long-term preference will weaken in [14], and time decay functions will fail in [16]. Therefore, the methods do not work well in the scenario, either. In this paper, our goal is *to identify viewpoints under low-level participation.*

Moreover, unlike a traditional medium, a microblog has some particular characteristics, such as short length, massive size, low quality, real-time nature and social networking. Thus, viewpoint discovery poses some challenges due to such characteristics under low-level participation. First, tweets are deficient in statistical and linguistic features due to short length. The existing methods for mining long texts are not suitable for microblogs. Second, tweets are very personalized as user-generated contents, which often take on complex styles, combine diverse information and embed much noise. Third, as user participation is often inactive in microblog events, *How to model user postings to handle such a situation?* Fourth, user viewpoints are usually unclear, and will change during a microblog event. *How to model user postings in order to capture them?* Finally, in the absence of the ground truth, the evaluation of mining viewpoints is not trivial.

To summarize, our main contributions are as follows:

- We design a framework for mining collective viewpoints in microblogs, and our proposed approach is language-independent in essence.
- We propose a Term-Retweet-Context (TRC) graph model which incorporates term co-existence and retweeting context information simultaneously to model retweeting so as to exploit viewpoints under low-level participation.
- In the absence of ground truth, we perform extensive experiments on a real microblog dataset to demonstrate the effectiveness of our approach.

The rest of the paper is organized as follows. Section 2 provides a review of related work. Section 3 introduces background and formulates the viewpoint discovery problem. Section 4 details mining viewpoints on TRC graph. Section 5 presents the experimental results on a real microblog dataset. Finally, our paper concludes in Section 6.

2   Related Work

In this section, we introduce related work on topic summarization and Random Walk methods.

Topic Summarization. The most closest work is topic summarization. Hierarchical summarization based on agglomerative clustering [15,17] is one of the most popular methods. Document clustering is applied to achieve topic summarization. Yang et al. [15] apply hierachical and non-hierarchical document clustering algorithms to a corpus of news stories, and focus on the exploitation of both content and temporal information. Zhao et al. [17] propose constrained agglomerative algorithms which combine the features of both partitional and agglomerative algorithms for text clustering. On the other hand, some other methods utilize sentences to achieve topic summarization. Hu et al. [5] aim to

extract representative sentences from a blog post which best represent the topics from its comments. Li et al. [8] propose a novel method that incorporates novelty, coverage and balance requirements into a sentence ranking probability model for producing summaries highly relevant to the topic. Note that multiple viewpoints actually may be involved in one tweet because of complicated human sentiments. Thus, these methods are not suitable for viewpoint summarization.

Since short texts have few semantic and statistical features, [6,2] improve the accuracy of short text clustering by enriching their representation with additional features from Wikipedia or WordNet. However, these methods are not suitable for Chinese microblog clustering for lack of reliable knowledge bases like WordNet.

Besides content information, [3,10] take structural information into account to achieve topic discovery. Carenini et al. [3] propose a new framework for email summarization, which utilizes clue words to combine the content and the structure of the quotation graph. Qamra et al. [10] propose a Content-Community-Time graph that can leverage the content of entries, their timestamps, and the community structure of the blogs, to automatically discover stories. However, the methods can't be directly applied to mining microblogs due to its characteristics, such as the style of user interaction and posting.

Graph Based Clustering. Graph clustering is an important technique in web mining and analysis. Random Walk is one of the most widely used proximity measurement in graph clustering. Some research work mainly focus on random walk algorithms itself. Tong et al. [12] aim to improve the efficiency of random walk algorithms for large graphs by exploiting two important properties, linear correlations and block wise community-like structure. Zhou et al. [18] propose a novel graph clustering algorithm, SA-Cluster, based on both structural and attribute similarities through a unified distance measure.

Also there are a lot of methods which can be used to exploit viewpoints. Xiang et al. [14] propose Session-based temporal graph to efficiently model users' long-term and short-term preferences for temporal recommendation. Zhao et al. [16] propose Term-Tweet-User graph, which simultaneously incorporates text content, community structure and temporal information, to model user postings over time for viewpoint discovery. However, these methods relying on high-level participation do not work well in a low-level situation.

3   Problem Statement

In this section, we will first introduce the background of retweeting and then give our problem definition.

3.1   Background

As a social network application, a microblog helps users to build relationships with others by retweeting. A retweet (*RT* for short) is a kind of quote and comment of someone's tweet. If you come across an interesting tweet, you can

republish it so that your followers can see it too. In this way, good ideas and thoughts can be spread rapidly in the social networks. A simple example in the Sina microblog's[3] style is shown below:

//Page:Great!//Gates:WOW!//Jobs:iPhone 4s is the most amazing iphone.

r_2 r_1 r_0: Retweet-Source

Fig. 1. An Retweet Example

Fig. 1 indicates that tweet r_0 is spread among three users. The tweet was originally posted by "Jobs" and then retweeted by his followers "Gates" and "Page" in order. Both r_1 and r_2 are the retweets of r_0, which is the source of all subsequent retweets, and r_1 links r_0 and r_2 together. Three users are chained up by retweeting. In a word, retweeting provides text content, social networking and context information.

3.2   Problem Definition

Problem 1. (Viewpoint Discovery)

Given: Towards a specific event, diverse viewpoints may emerge. Each viewpoint represents user opinions and thoughts. A microblog retweeting dataset on a given event contains such viewpoints under low-level participation.

Goal: Each viewpoint can be depicted by a few representative terms. We aim to identify those collective viewpoints by grouping representative terms.

Usually users posted a large amount of retweets during a hot event. They discussed with their followers by retweeting and propagated their opinions and thoughts. Since retweeting messages can provide not only text content, but also social networking and semantic context information, we prefer retweets as microblog datasets. However we observe that most users often post only one tweet during a hot event. User participation is not so active as imagined. And in most real applications social networks are very sparse and weak due to low-level participation. Therefore, the key problem is *how to mine collective viewpoints under low-level participation?* In this paper, our key idea is to utilize retweeting context information, which is not influenced by participation levels, to model retweeting for viewpoint discovery. In the next section, we will detail our proposal.

4   Mining Viewpoints on TRC Graph

In this section, we will illustrate how to mine collective viewpoints based on TRC graph. First, we present an overview of our proposed framework. Then we describe how to model retweeting by means of context information and construct TRC graph. Finally, we apply Random Walk on the graph to measure term relevance and group representative terms into viewpoints.

---

[3] http://www.weibo.com

4.1 The Framework of Mining Viewpoints

Our proposed framework consists of three consecutive phases, including *Microblog Retweet Pre-processing*, *TRC Graph Mining* and *Viewpoint Generation*, as shown in Fig. 2.
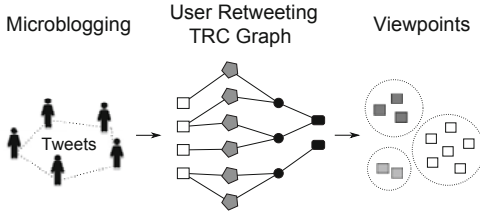

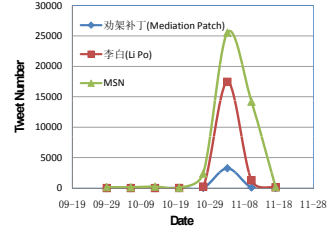**Fig. 2.** Framework for Viewpoint Discovery

Fig. 3. Distribution of Representative Terms

Microblog Retweet Pre-processing Phase. We design and develop a microblog crawling program, which posts keyword queries to a microblog search engine and collects search results (i.e., tweet web pages), extract representative terms through an entropy method, and build retweet context relationships by analysing retweeting interactions.

TRC Graph Mining Phase. Our approach leverages retweet content and context information to construct the *TRC* graph, as shown in Fig. 4. And then Random Walk is posed on the graph in order to measure each term correlation with other ones.

Viewpoint Generation Phase. With respect to each term, we obtain a ranking of correlations with other ones after the previous phase. Term clustering based on shared nearest neighbour (SNN) is performed. The result clusters are represented as all kinds of viewpoints.

4.2 Pre-processing: Representative Term Selection

In real applications, retweets include various terms as user-generated contents. But these terms are not equally important and useful. Only representative terms can enhance the performance of viewpoint discovery. Therefore, we should identify representative terms in the term set. It is observed that representative terms are usually associated with frequency bursts in a certain period. Leveraging the characteristic can improve the quality of final viewpoints.

Usually TF-IDF [9] is widely used for feature selection in textural corpora and also can be used to extract representative terms. In practice representative terms can not be effectively distinguished from non-representative ones through the methods based on TF-IDF, because the technique neglects term distributions. Fig. 3 shows representative terms usually have sudden burst of frequency in some period, in other words, representative terms are not uniformly distributed.

In this way, we utilize *Entropy* to find the terms with a frequency burst. The entropy of a term t is defined as $Entropy(t) = -\sum_{i=1}^{k} p_i * log(p_i)$. We count the frequencies (TF_i in the ith day) of term t in the k days. The probability p_i is TF_i/TF, where TF is the total term frequency during the event. The terms whose entropy is smaller than a specified threshold will be selected.

4.3 Modelling Retweeting on TRC Graph

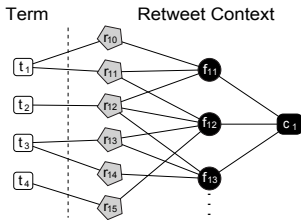In this section, we will illustrate how to model retweeting by means of different kinds of objects.
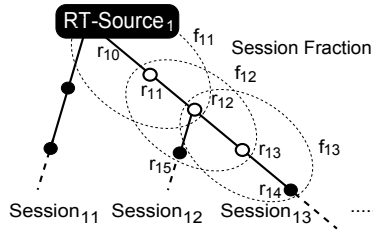

Fig. 4. Graph Representation   **Fig. 5.** An Example of Retweet Cluster

4.3.1 Retweeting Context

Definition 1. (PATH) *A path in a graph is a sequence of nodes such that from each of its nodes there is an edge to the next nodes in the sequence. In this paper, we assume that there is no repeated nodes in paths.*

During hot events we can collect a large amount of retweets R, which can be grouped into retweet clusters $C = \bigcup_i C_i$ according to different retweet sources. Each retweet cluster C_i contains a set of retweets r_{ij}, and r_{i0} is the source of all retweets. Thus, $R = \bigcup_i C_i = \bigcup_i \bigcup_j r_{ij}$.

For example, Fig. 5 shows that a retweet cluster C_1 contains a set of retweet sessions $C_1 = \bigcup_j S_{1j}$. Each session is a *path* in graph theory. It is like the retweeting sequence in Fig. 1. Different sessions may have a common subsequence, that is, they may originate from the same retweet source.

Fig. 5 also indicates that each retweet lies in a corresponding session context, which is a session fraction. For example, session fraction of r_{12} is $f_{12} = \{r_{11}, r_{12}, r_{13}, r_{15}\}$ Retweet r_{12} is adjacent to retweet r_{11}, r_{13} and r_{15}. The closer two retweet nodes are in a path, the greater the relevance between them is. Therefore, there is a high probability that the retweets in the same session fraction are more relevant.

4.3.2 TRC Graph Construction

In this section, we will construct a TRC graph for modelling retweeting in Fig. 4. The graph $G = (V, E)$ is a finite and node-labeled 4-partite graph. V may be divided into disjoint partite sets. Each edge in E connects two nodes from different partite sets; that is, there is no edge between two nodes in the same partite

set. The graph G includes four types of objects: terms, retweets, session fractions and retweet clusters, which are denoted by node set T, R, F and C respectively. $|\cdot|$ denotes the element number of a set. Node t,r,f and c denote four types of objects respectively. There are also three kinds of relations between different node sets, which are denoted by edge set $e(t,r)$, e(r,f) and $e(f,c)$ respectively. Edge $e(t,r)$ represents that term t occurs in tweet r. Edge $e(r,f)$ represents that retweet r belongs to session fraction f. And edge $e(f,c)$ represents that session fraction f belongs to retweet cluster c. All edge weights of TRC graph are set to be 1. Finally, we obtain a TRC graph for mining microblogs.

4.4   Random Walk on TRC Graph

Random Walks with restart will be applied on TRC graph for proximity measurement. The equation is defined as $\boldsymbol{h_i} = \gamma\boldsymbol{h_i}\mathbf{P}^N + (1-\gamma)\boldsymbol{e_i}$, where $\boldsymbol{h_i}$ represents the proximity vector w.r.t. node t_i, $\mathbf{P}^N$ is the normalized matrix of a weighted matrix $\mathbf{P}$, $1-\gamma$ is the restart probability for random walks, $\boldsymbol{e_i}$ is a starting vector, the ith element 1 and 0 for others.

There are two widely used ways ($PreCompute$ and $OnTheFly$) to solve random walks with restart [12]. $PreCompute$ method requires pre-computation and is suitable for queries of node proximity in the on-line response time. $OnTheFly$ method does not require pre-computation and additional storage cost. Its response time depends on the iteration number and the number of edges. Therefore, $OnTheFly$ method is more suitable especially when the graph is massive. In this paper, we prefer $OnTheFly$ method for computing the relevance scores of nodes.

Algorithm 1 gives the overview of our proposed method. We first construct the weighted matrix $\mathbf{P}$ of TRC model from the original raw dataset (step 2). A symmetric matrix $\mathbf{P}$ consists of three block matrixes: $Term\text{-}Reweet$ block matrix , $Retweet\text{-}Session\_Fraction$ block matrix and $Session\_Fraction\text{-}Session\_Cluster$ block matrix. Then (step 3), we normalize matrix $\mathbf{P}$. Finally (step 4-8), Random Walk will be iterated until convergence. In the algorithm, convergence should occur as the weight distribution of edges does not change in each iteration.

In Algorithm 1, there are several ways to normalize the weighted matrix. Our algorithm uses row normalization which is a most natural method to normalize matrix $\mathbf{P}$. Correspondingly, we take the normalized matrix $\mathbf{P}^N$ as the input.

5   Experiments

5.1   Dataset Description

We crawled a real dataset from Sina microblog, which has been developing rapidly and greatly in China. According to Sina, it has more than 50 million users, and increases by 10 million new users per month in 2010. We collected 5,831 retweet clusters, which contain 23,834 messages about "Expo 2010 Shanghai China" event posted by 15,162 users. The dataset was collected through Sina

---

Algorithm 1. Computing the Proximity Vectors of Terms

Input: TRC Graph Dataset
Output: Term Ranking Matrix **H**

1 $n \leftarrow |V|, m \leftarrow |T|$;
2 Construct adjacent matrix **P** of TRC graph ;
3 Normalize **P** to obtain $\mathbf{P}^N$;
4 **foreach** $term\ t_i \in T$ **do**
5 | $h_i \leftarrow e_i$;
6 | **repeat**
7 | | $h_i \leftarrow \gamma s_i \mathbf{P}^N + (1 - \gamma)e_i$;
8 | **until** $convergence$;
9 $\mathbf{H} = [h_i]_{m \times 1}$;
10 **return H**;

---

microblog search engine. The duration of our dataset is from May 1st to May 31th in 2010. Table 2 shows the statistics of user participation during the event. In the next sections, we will demonstrate the effectiveness of our proposed methods.

Table 2. Statistics of Participation

# times	Percent of Users
1	78.9%
2	12.9%
3	3.7%
≥ 4	4.5%

5.2   Evaluation Metric

Since no ground truth is available for microblog datasets, we evaluate the quality of clustering results by means of *cohesion*. Given a set of clusters $C = \cup_i^n C_i$, we measure the cohesion using $Cohesion(C) = \sum_{i=1}^n \omega_i Cohesion(C_i)$, where ω_i is the weight of C_i. $Cohesion(C_i) = \sum_{x \in C_i y \in C_i} Proximity(x, y)$. The overall cluster cohesion $Cohesion(C)$ is the weighted sum of all cluster cohesions.

We measure the cohesion of the individual cluster in terms of the proximity of pairwise terms. In the absence of reliable semantic knowledge bases like Word-Net, we quantify the proximity of terms through *pointwise mutual information* (PMI). The PMI formula is defined as: $PMI(x, y) = log(\frac{p(xy)}{p(x)p(y)})$. The more relevant the terms are, the larger the PMI is. Therefore, the higher cohesion values are preferred.

5.3   Comparison Methods

In order to demonstrate the effectiveness of our proposed method ($RWTRC$), we design two algorithms: random walk based on Term-Retweet bipartite graph

(RWB) and random walk based on Term-Retweet-User graph ($RWTWU$) [16], to compare with $RWTRC$ in our experiments. The results of RWB and $RWTWU$ are used as baselines to examine whether retweet context information can improve mining performance at a low level of participation.
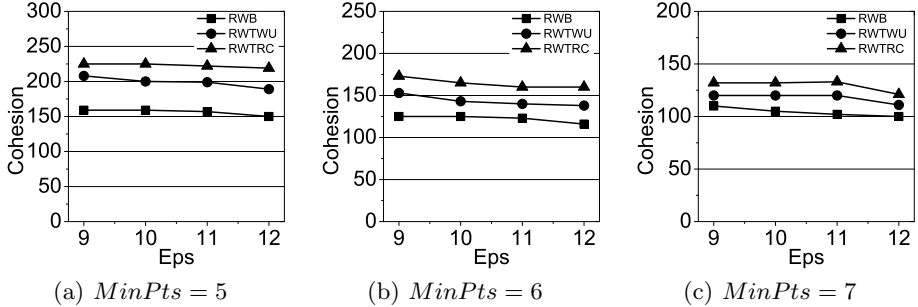

Fig. 6. Cohesion of RWB, $RWTWU$ and $RWTRC$ with different Eps

In *Microblog Retweet Pre-processing* phase in section 4.1, we extract 488 representative terms from the event dataset, which are included in TRC graph. In *Viewpoint Generation* phase, $DBSCAN$ clustering algorithm with shared nearest neighbour (SNN) is adopted. We evaluate the cohesions of RWB, $RWTWU$ and $RWTRC$ by setting different $DBSCAN$'s parameters: Eps and $MinPts$. In our experiments, Eps is SNN similarity, which is set from 9 to 12, and $MinPts$ is set from 5 to 7. Fig. 6 shows that our algorithm $RWTRC$ outperforms RWB and $RWTWU$ in all cases.
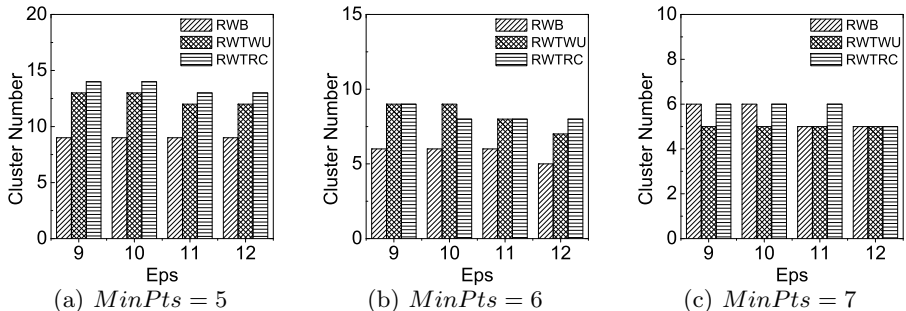

Fig. 7. Cluster Number of RWB, $RWTWU$ and $RWTRC$ with different Eps

Consider an extreme case, if every cluster contains two terms being most relevant, the overall cohesion must be the highest. But too many viewpoint clusters may be generated, and such experimental results are not meaningful. Therefore, we should examine the quantity of clusters besides the cohesion. Fig. 7
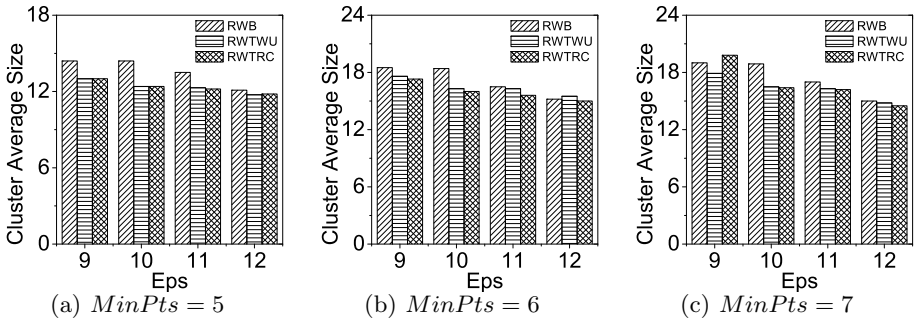
Fig. 8. Cluster Average Size of RWB, $RWTWU$ and $RWTRC$ with different Eps

and 8 show the statistics of clusters generated by three algorithms with different parameters. The number and size of clusters generated by all algorithms are appropriate and rational. This proves that cohesion is an effective evaluation of three algorithms in our experiments.

Finally, our experiment results show that RWB utilizing the simple feature like term co-existence has a worst performance. Although $RWTWU$ leverages more characteristics like temporal aspect and community structure, such characteristics are unavailable and the model should be reduced at a low level of participation. Compared to the other algorithms, $RWTRC$ considers the effective feature like retweeting context under low-level participation. Thus, $RWTRC$ achieves the satisfactory performance in this case.

6   Conclusions

In this paper, we propose a novel approach to exploit collective viewpoints in microblogs. To deal with low-level participation, we propose a graph-based model named Term-Retweet-Context (TRC), which incorporates retweet text content and context, to model user retweeting. We apply Random Walk on TRC to measure the relevance between representative terms and group them into collective viewpoints. The extensive experiments on the real dataset confirm the effectiveness of our proposed method.

References

1. Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., Aly, M.: Video suggestion and discovery for youtube: taking random walks through the view graph. In: WWW (2008)

2. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering short texts using wikipedia. In: SIGIR (2007)
3. Carenini, G., Ng, R.T., Zhou, X.: Summarizing email conversations with clue words. In: WWW (2007)
4. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: KDD (2004)
5. Hu, M., Sun, A., Lim, E.-P.: Comments-oriented blog summarization by sentence extraction. In: CIKM (2007)
6. Hu, X., Sun, N., Zhang, C., Chua, T.-S.: Exploiting internal and external semantics for the clustering of short texts using world knowledge. In: CIKM (2009)
7. Li, M., Dias, M.B., Jarman, I.H., El-Deredy, W., Lisboa, P.J.G.: Grocery shopping recommendations based on basket-sensitive random walk. In: KDD (2009)
8. Li, X., Shen, Y.-D., Du, L., Xiong, C.-Y.: Exploiting novelty, coverage and balance for topic-focused multi-document summarization. In: CIKM (2010)
9. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
10. Qamra, A., Tseng, B.L., Chang, E.Y.: Mining blog stories using community-based and temporal clustering. In: CIKM (2006)
11. Su, Q., Xu, X., Guo, H., Guo, Z., Wu, X., Zhang, X., Swen, B., Su, Z.: Hidden sentiment association in chinese web opinion mining. In: WWW (2008)
12. Tong, H., Faloutsos, C., Pan, J.-Y.: Fast random walk with restart and its applications. In: ICDM (2006)
13. Turney, P.D.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: ACL (2002)
14. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long-and short-term preference fusion. In: KDD (2010)
15. Yang, Y., Pierce, T., Carbonell, J.G.: A study of retrospective and on-line event detection. In: SIGIR (1998)
16. Zhao, B., Zhang, Z., Gu, Y., Gong, X., Qian, W., Zhou, A.: Discovering Collective Viewpoints on Micro-blogging Events Based on Community and Temporal Aspects. In: Tang, J., King, I., Chen, L., Wang, J. (eds.) ADMA 2011, Part I. LNCS, vol. 7120, pp. 270–284. Springer, Heidelberg (2011)
17. Zhao, Y., Karypis, G.: Evaluation of hierarchical clustering algorithms for document datasets. In: CIKM (2002)
18. Zhou, Y., Cheng, H., Yu, J.X.: Graph clustering based on structural/attribute similarities. PVLDB, 2(1) (2009)

Leveraging Network Structure for Incremental Document Clustering

Tieyun Qian[1,3], Jianfeng Si[2], Qing Li[2], and Qian Yu[1]

[1] State Key Laboratory of Software Engineering,
Wuhan University, Wuhan, China
[2] Department of Computer Science,
City University of Hong Kong, Hong Kong, China
[3] State Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing, China
qty@whu.edu.cn, jianfsi2@student.cityu.edu.hk
itqli@cityu.edu.hk, ccdyuqian@yahoo.cn

Abstract. Recent studies have shown that link-based clustering methods can significantly improve the performance of content-based clustering. However, most previous algorithms are developed for fixed data sets, and are not applicable to the dynamic environments such as data warehouse and online digital library.

In this paper, we introduce a novel approach which leverages the network structure for incremental clustering. Under this framework, both the link and content information are incorporated to determine the host cluster of a new document. The combination of two types of information ensures a promising performance of the clustering results. Furthermore, the status of core members is used to quickly determine whether to split or merge a new cluster. This filtering process eliminates the unnecessary and time-consuming checks of textual similarity on the whole corpus, and thus greatly speeds up the entire procedure. We evaluate our proposed approach on several real-world publication data sets and conduct an extensive comparison with both the classic content based and the recent link based algorithms. The experimental results demonstrate the effectiveness and efficiency of our method.

1 Introduction

With the development of the Internet, the number of online publications has grown exponentially in recent years. For example, in the field of computer science, DBLP, a computer science bibliography site, has indexed more than one million articles and is still growing at a rapid pace. Due to the huge number of publications, it has become a very challenging task to manage, organize and present the relevant documents for the users.

Document clustering is a traditional technique which helps structure documents into categories for better understanding. Early work mainly utilizes a vector space model to represent the textual information and some metrics like cosine similarity are used to measure the distance between two documents. Due to the high dimensionality of the term space and the large size of the corpus, the above text-based clustering algorithms are often of less efficiency and lower clustering quality.

Besides the main contents, most documents contain linkage information. For example, scientific papers often have references and web pages are linked to other pages.

The texts and the links of documents together form a document network. A number of algorithms have been proposed for clustering the documents in a document network by employing both content and linkage information, such as neighborhood based clustering [1], scalable community discovery [7], and relaxing labeling based clustering [15]. A thorough comparative evaluation performed by Zhang et al. demonstrates that their link-based method exhibits significant improvements over content-based clustering. However, this approach, like most of the other link-based methods, is only developed for static data sets, and is not applicable to the dynamic environments such as data warehouse and online digital library, where the documents are always received incrementally.

To well meet the need of managing the ever increasing publications, we present a novel framework which leverages the network structure for incremental document clustering in this paper. Our method has two unique properties. Firstly, both the content and linkage information are incorporated to determine the membership of a cluster. The combination of these two types of information ensures that the proposed approach can achieve promising results. Secondly, the status of core members (those nodes with larger degree) is used to quickly determine whether to split or merge a new cluster. This filtering process can greatly speed up the entire procedure by eliminating many unnecessary and time-consuming checks on the whole corpus. The effectiveness and efficiency of our method are demonstrated by the evaluation on several different types of real world data sets.

2   Related Works

There is a rich literature on document clustering and a daunt number of algorithms have been proposed. The methods are categorized into three principal themes, i.e., generative techniques [8,11], discriminative [13,17], and spectral methods [4]. The generative algorithms assume that there exists a distribution of the corpus and then use an iterative procedure like EM to estimate the model. The main problem with these types of methods is the high time complexity. The discriminative algorithms define an objective function and aim to optimize this criterion based on the pair-wise similarity. The spherical k-means [3] method is the most popular yet one of the most efficient discriminative algorithm, although it has some drawbacks such as being sensitive to the initialization and covering to the local minima. Interpreting the adjacent matrix on similarity as a graph, the spectral algorithms produce the clusters by finding cuts in the graph. Spectral approaches are in general computationally demanding and it is infeasible to compute the eigenvectors for large graphs.

The above studies rely on the content of documents. With the growing number of textual data with relations, there have been some discussions [15,14,9,1,7] regarding link based approaches in document clustering. This category of clustering methods utilizes the object adjacency relations to determine the class label of a document. For example, Angelova et al. applied a relaxation labeling approach to improve the traditional text clustering [1]. As shown by literature studies [15], these link based methods are effective in enhancing text based document clustering. Nonetheless, this type of framework requires an iterative procedure which prevents it from being scalable to large data collections. Recently, Li el al. [7] develops a scalable community discovery solution for

large scale textual datasets. Whereas this method provides valuable insights on employing the core members in community discovery, it does not deal with the splitting and merging phenomena in the course of dynamic clustering.

In general, the batch mode methods, either text or link based, are too time-consuming to be appropriate for realtime and dynamic applications. It has been shown that the incremental approaches are more effective than the batch ones, and some incremental clustering [5,18,10] have been introduced in the literature. However, the existing incremental clustering algorithms mainly focus on the flat document. They do not investigate how the link information can be integrated into the online framework. Furthermore, the effects of network structure for incremental clustering have not been evaluated on various types of links.

3   Preliminaries and Problem Statement

We first introduce some definitions that will be used in the following context.

Definition 1. *Document.* *Let $V = \{d_1, d_2, ..., d_n\}$ be a document collection with a vocabulary $T = \{w_1, w_2, ..., w_m\}$. Each document d_i in V consists of a bag of words and is represented with a vector $d_i = \{ w_{i1}, w_{i2}, ..., w_{im} \}$, where w_{ij} is the weight of each term w_j in document d_i.*

Definition 2. *Document Network.* *Let $G = (V, E)$ be an undirected graph representing the document network with vertex set V and edge set E. Each node d_i in V is a document. There exists an edge e_{ij} in E if two nodes d_i and d_j are adjacent.*

Definition 3. *Neighborhood.* *The neighborhood of a document d_i, denoted as $\Gamma_{di} = \{d_j | e_{ij} \in E\}$, is a set of nodes with links connected with d_i.*

Definition 4. *Degree.* *The degree of a node d_i is the size of its neighborhood, i.e. $deg(d_i) = |\Gamma_{di}|$.*

Definition 5. *Clustering.* *Given a document network $G = (V, E)$, the problem of clustering consists of dividing the vertices in G into k disjointed partitions $C_1 = (V_1, E_1), C_2 = (V_2, E_2), \ldots, C_k = (V_k, E_k)$ such that $V_i \bigcap V_j = \phi$ $(i \neq j)$, and $\bigcup_{i=1}^{k} V_i = V$.*

Definition 6. *Core Set.* *A document d_i is defined as a core member if its degree ranks high among all nodes(according to a predefined parameter: $core\_percent \in (0, 1]$). The core set K_i of a cluster C_i is a set of core members.*

Definition 7. *Cluster Centroid.* *For a document network $G = (V, E)$ and a cluster distribution $C = \{C_1, C_2, .., C_k\}$ on G, the centroid of each cluster C_i is represented as the average of its core members, which is defined as $\overrightarrow{C_i} = \frac{\sum_{j \in K_i} \overrightarrow{t_j}}{|K_i|}$, where t_j is the text vector of d_j.*

Here the core set is used to represent the centroid of a cluster. By maintaining the centroid over the core members, the number of updating operations will greatly decrease since they only need to be done when the cores changed.

Definition 8. *Incremental Clustering. Given a document network $G = (V, E)$, an existing cluster distribution $C_1 = (V_1, E_1), C_2 = (V_2, E_2), \ldots, C_k = (V_k, E_k)$ and a new document d_x, the problem of incremental clustering is to assign d_x to an existing C_i such that a predefined metric is optimized.*

A typical incremental clustering algorithm consists of three steps. (1) Finding the host cluster of a new arriving document; (2) Checking whether to change the larger cluster, i.e. should it be split or merged; (3) Determining the membership of a cluster when the change actually occurs. The first and the third step are critical to the clustering quality, while the most time-consuming procedure lies in the second step.

4   Incremental Clustering on Document Network

We make use of the linkage information by two means. Firstly, for the efficiency purpose, we propose to use the core set of the network as a filtering condition during the second step of an incremental clustering. The rationale behind this approach is that many unnecessary check operations can be pruned with very small overhead. Secondly, for the efficacy purpose, we integrate the information from neighborhood to decide the host cluster of a new document or the membership of the cluster to be split. The key idea is that the immediate neighbors are more likely to join the same cluster.

4.1   The Overall Framework

Suppose we have already got an initial cluster distribution, the task of incremental clustering is to assign a label to the new document and then decide whether to change the cluster. We first outline the overall framework in Algorithm 1, the technique details are introduced in the following parts.

Algorithm 1. Linked-IC

Input: Current cluster distribution, a new document d_x, the neighborhood relationship between d_x and existing documents
Output: d_x's cluster assignment
Steps:

1. Compute the probability of assignment of d_x to each Cluster C_i based on text similarity.
2. Compute the probability of assignment of d_x to each Cluster C_i based on its neighbor's cluster labels.
3. Combine above probabilities into final score, and assign d_x to cluster C_i with the largest possibility. Update C_i's centroid when new core members are introduced.
4. After a certain number of new documents have been added, check the condition for splitting for each cluster C_i and merging for each pair of clusters (C_i, C_j).
    a **if** $CheckSplit(C_i) = $ **True**
      then $DoSplit(C_i, subcluster1, subcluster2)$
    b **if** $CheckMerge(C_i, C_j) = $ **True**
      then $DoMerge(C_i, C_j)$

In Algo. 1, steps 1-3 are used to assign the new document to a cluster, while step 4 decides the membership of a cluster.

4.2   Assign a New Document to Existing Clusters

Given a new coming document d_x, its text information t_x, its neighborhood Γ_x, and an initial clustering distribution $C = \{C_1, C_2, .., C_k\}$, the assignment of d_x to C_i consists of the following three main steps.

1. The computation of text impact

$$P(C_i|t_x) = \frac{\cos(\overrightarrow{t_x}, \overrightarrow{C_i})}{\sum_{j=1}^{K} \cos(\overrightarrow{t_x}, \overrightarrow{C_j})}, \tag{1}$$

   Where t_x is represented by the TF-IDF based vector space model (VSM) [12], and $\boldsymbol{C_i}$ is the cluster centroid defined in Definition 7.

2. The computation of link impact

$$P(C_i|\Gamma_x) = \frac{|\{d_j | d_j \in \Gamma_x, l(d_j) = i\}|}{|\Gamma_x|}, \tag{2}$$

   Where $l(d_j)$ is the cluster label of the neighbor document d_j. As the neighbors often intend to take the same cluster assignment, the information from immediate neighbors is employed to calculate the impact of network structure.

3. Combination

$$P(C_i|d_x) = w * P(C_i|t_x) + (1 - w) * P(C_i|\Gamma_x). \tag{3}$$

   The host cluster of d_x is finally chosen by the linear combination of text and link impacts. Here $w \in [0, 1]$ is a weight for two factors. For example, by setting $w = 0$, we ignore the effects of content, and the algorithm reduces to a pure link-based clustering method.

4.3   Check the Condition for Changing

As the new documents continuously joining, the cluster distribution may change. Some small clusters become more closely related and they may merge into a bigger one. An old topic gestates a new topic and it may split into two subclusters. To ensure that the cluster distribution changes in accordance with the dynamic of the topic development, we need to check whether to split one cluster or to merge two clusters. The basic rationale is that: (1) When the inner dissimilarity of a cluster enlarges, the cluster should split. (2) When the similarity between two clusters are large enough, the two clusters should merge into one.

A traditional content-based incremental clustering is often computational expensive, as it requires to check the average similarity rather frequently to see whether the cluster distribution has changed. Fortunately, in a document network, such a method can be greatly improved with the help of linkage information. Intuitively, a small change in the fringe of a graph should not bring about an immediate disturbance on clusters. Only when the core members of a cluster change a lot, there is a necessity to further check the condition for splitting. Hence the status of the core set can be used as a filtering condition. For example, a split may occur to a cluster when the number of new cores has reached a certain proportion. Only when such a condition is satisfied, will a further comparison be performed to check the text similarity. Algorithm 2 illustrates such a procedure.

Algorithm 2: CheckSplit

Input: Cluster C_i to be checked and its core set K_m and K_n, where K_m is the old set in C_i and K_n is the newly formed one.

Output: True or False

Steps:

1. Check the proportion of the number of new cores to that of the old ones, return False directly when the following condition fails. Otherwise go to the next step.

$$\frac{sizeof(K_n)}{sizeof(K_m)} > \tau \tag{4}$$

2. Compare the average text similarity between two clusters based on their core sets, return True when the following condition is satisfied.

$$Min(avgSim(K_m), avgSim(K_n)) > avgSim(K_m \cup K_n) \tag{5}$$

3. return False

In Algorithm 2, $avgSim$ is the cosine similarity taken over the TF-IDF based vector of the document content. K_m represents the old core set of an existing cluster, while K_n represents the core set of a new cluster after a time period.

Similar to the process for splitting, when the merging operation is conducted, we first check the cutting edge rate and then check the text similarity between two clusters. The checking on graph structure helps filter out unnecessary text similarity computation and thus greatly speeds up the procedure. The procedure is described in Algorithm 3.

Algorithm 3: CheckMerge

Input: Two clusters C_i and C_j and their core sets K_m and K_n

Output: True or False

Steps:

1. Check the cutting edge rate between two clusters, return False directly if the following condition fails. Otherwise go to the next step.

$$Min(\frac{cutNum(K_m, K_n)}{intraEdge(K_m)}, \frac{cutNum(K_m, K_n)}{intraEdge(K_n)}) > \tau \tag{6}$$

2. Compare the average text similarity between two clusters based on their core sets, return True when the following condition is satisfied.

$$Max(avgSim(K_m), avgSim(K_n)) < avgSim(K_m \cup K_n) \tag{7}$$

3. return False

In Algorithm 3, $cutNum$ represents the number of cut edges between two clusters, while $intraEdge$ represents the number of inner edges inside one cluster.

Compared with the previous incremental clustering methods, our proposed algorithm is very efficient due to two reasons. Firstly, a filtering step which well utilizes the graph information is performed before the further action is taken. Secondly, unlike computing text similarity over all documents, we use the core set as the representatives of one cluster. This also improves efficiency and helps rule out side-effects from outliers.

4.4   Update the Clustering Membership

The merging of two clusters C_i and C_j is quite simple. We can easily form a large cluster to contain the vertices in C_i and C_j. Hence we omit the detail here.

The splitting process of a cluster C_i is more difficult. Supposing there are n nodes in C_i, the number of partitioning would be $2^n - 1$. In this paper, we adopt the following strategy to efficiently split C_i into two sub-clusters. Firstly, we use the weighted furthest and most dissimilar pair of cores to initialize the two sub-clusters. Secondly, we assign the remaining members to sub-clusters based on Equations (1)-(3).

The furthest core pair is defined as a pair of two cores with the smallest number of their joint neighbors, i.e., co-occurrence.

$$pairSet1 = \{(d_i, d_j)|$$
$$(d_i, d_j) = argMin_{(d_i \in K_m, d_j \in K_n)}(co - occurrence(d_i, d_j))\}. \tag{8}$$

If there are more than one pairs with the same value of common neighbors, we choose the one with the maximum sum of degrees of two cores.

$$pairSet2 = \{(d_i, d_j)|(d_i, d_j) \in pairSet1,$$
$$(d_i, d_j) = argMax_{(d_i \in K_m, d_j \in K_n)}(deg(d_i) + deg(d_j))\}. \tag{9}$$

In addition, we use the most dissimilar pair to enlarge the initial set of documents in subclusters, which is defined as:

$$pairSet3 = \{(d_i, d_j)|$$
$$(d_i, d_j) = argMin_{(d_i \in K_m, d_j \in K_n)}(cos(\overrightarrow{t_i}, \overrightarrow{t_j}))\} \tag{10}$$

Overall, the weighted furthest pair is from the view point of a graph, while the most dissimilar pair is based on the content. Thus both the link and text information contribute to the initialization of subclusters. The detail is shown in Algo. 4.

Algorithm 4: DoSplit

Input: Cluster C_i to split
Output: subcluster1 and subcluster2
Definition:
Steps:

1. Find the initial centroids $pairSet = \{(d_i, d_j)\}$ for two sub-clusters.
   if $|pairSet1| = 1$
   then $pairSet = pairSet1 \bigcup pairSet3$
   else $pairSet = pairSet2 \bigcup pairSet3$

2. Initialize two sub-clusters with core documents in pairSet.
   $initCluster(subcluster1, d_i);$
   $initCluster(subcluster2, d_j)$

3. Assign remaining documents into sub-clusters respectively according to Equations 1-3.

---

5   Evaluation

5.1   Experiment Setup

In order to evaluate our method, we conduct experiments on three real-world data sets with basic link types, i.e., CORA7 with citation relations, WIKI7 with hyperlink relations, and DBLP3 with co-authorship relations. These data sets have also been used in previous studies like [15], [6], and [2]. Their class labels are already known.

The general statistics for these data sets are shown in Table 1.

Table 1. Data sets

Dataset	# of classes	# of docs	avg degree	# of added docs
CORA7	7	4263	5.31	1371
WIKI7	7	5360	15.66	1007
DBLP3	3	16809	19.68	6887

For each dataset, we remove documents from a certain class and gradually add them back into the collection as a simulation of the incremental environment. After every 10% of these documents are added back to the collection, we output the current data collection and run the Content based K-Means(CBKM) and Link based K-Means(LKKM) on the collection in batch mode, and then compare the results with that of our Incremental Clustering algorithm(IC). Since our method also employs the link information, we carry out these comparisons to see how the network structure can be used to enhance the incremental document clustering.

The cluster quality is evaluated by two commonly used metrics: Purity[16], and Normalized Mutual Information [19] (NMI).

– Purity

$$Purity = \frac{1}{n} \sum_{i=1}^{K} Max(n_{i1}, \cdots, n_{iC})$$

– NMI

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} = \frac{\sum_{i=1}^{K} \sum_{j=1}^{C} n_{ij} log(\frac{n*n_{ij}}{n_i*n_j})}{\sqrt{(\sum_{i=1}^{K} log(\frac{n_i}{n}))(\sum_{j=1}^{C} n_j log(\frac{n_j}{n}))}}$$

In above equations, n is the total number of documents, n_{ij} is the member size of class j in cluster i. n_i is the member size of cluster i and n_j is the member size of class j. X is the cluster label variable for cluster assignments while Y is the actual class label variable. K represents the cluster number while C represents the class number.

5.2   Thresholding and Scaling Effects

A key property of our framework is the utilization of the core set. They are used as the representatives of clusters, and also critical to the filtering process. An ordinary way to find the key nodes in a graph is to set a threshold for degree. A node is identified as a core if its degree is greater than the predefined threshold. However, such a threshold

is data dependent and is very difficult to tune. In practice, we select the core members by using another parameter $core\_percent \in (0, 1]$ instead. This parameter reflects the proportion of core members to all nodes and is more flexible than the absolute threshold of degree. The value of $core\_percent$ will have a direct impact on the clustering quality, and also on the efficiency of the algorithm. In order to reveal their relationship, we conduct experiments on three datasets under $core\_percent$ changing from 0.1 to 1 by step 0.1. The results are shown in Fig.1
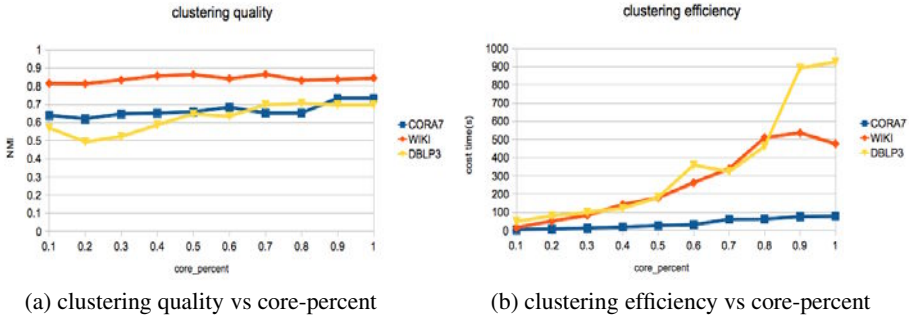

(a) clustering quality vs core-percent   (b) clustering efficiency vs core-percent

Fig. 1.

From Fig.1(a), we can find that the smaller proportion of core members will lower the clustering quality a bit. Nevertheless, it does not affect the quality too much. In other words, the small size of core set can still well represent the whole data collection. Furthermore, from Fig. 1(b), we can see that the increase of overhead on CORA is the least among three datasets. This is due to the smaller number of documents and the simpler network structure of this dataset. Remember the average degree of CORA is only 5.31, nearly $1/3$ and $1/4$ to that of WIKI and DBLP. On the whole, the time cost grows almost linearly with the increase of the $core\_percent$ on all the three data sets. When combining Fig.1 (a) and (b) together, we can further draw the conclusion that a relatively small value of $core\_percent$ sacrifices a little clustering quality but gains a great improvement on efficiency.

We also carry out comparisons of the proposed algorithm under various settings of other two parameters, i.e., w (the text weight in Equation 3) and τ (the filtering threshold in Equations 4 and 6). The results show that the clustering quality reaches the highest when w changes around 0.7 and τ around 0.1. According to the experimental results, we set the $core\_percent = 0.5$, $w = 0.7$, and $\tau = 0.1$ for the following experiments.

5.3   An Example of Cluster Splitting

As new documents enter the collection, the cluster distribution gradually changes. Our proposed incremental clustering algorithm is able to capture this tendency and adjust the results in the course of time. Fig.2 depicts a splitting phenomenon found by our experiment on CORA7 dataset. When a small number of documents arrive, some of them are first attached to the original cluster. With the joining of more members, a new topic gradually forms, and the border between the old and the new cluster becomes much clear. Finally, the new cluster is broken away from the old one.
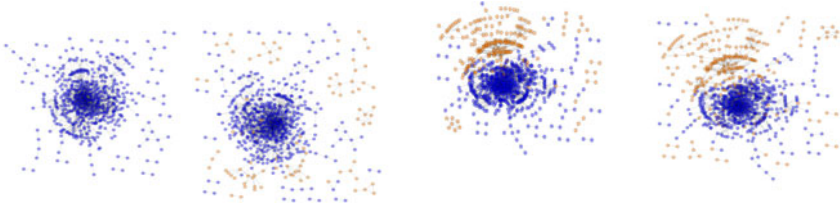
Fig. 2. An example of cluster splitting on CORA7

5.4   Comparison of Clustering Efficiency

The main motivation of our study is the scalability of clustering method under the dynamic environments like data warehouses which contain a great number of publications. We conduct the comparative studies of CBKM, LKKM and our proposed IC method on three real data sets. Fig.3 shows the time cost when the incremental ratio of documents changes from 0.1 to 1 by step 0.1.

From all the three figures, we can observe a huge improvement of IC over CBKM and LKKM on clustering efficiency. The performance of LKKM is the poorest due to the multiple iterations it needs to re-estimate the probability. Compared with our incremental clustering method, the time cost for the pure content based approach is also very high. This implies that even though we integrate the linkage information, the time overhead does not increase too much. From Fig. 3 (c), the advantage of IC becomes more evident. This reveals that IC is much more appropriate for the data collection with a larger size and more complicated structure like DBLP.
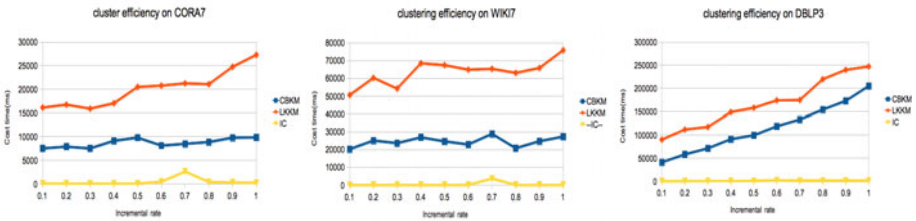

Fig. 3. Comparison of clustering efficiency on (a) CORA, (b) WIKI, (c)DBLP

5.5   Comparison of Clustering Quality

We examine the quality of clustering approaches in Table.2, Table.3, and Table.4.

It can be seen from Table 2 and Table 3 that, the improvement of IC over CBKM is evaluated significantly on CORA and DBLP data sets by two metrics-NMI and Purity. From these tables, we also observe that IC outperforms LKKM by Purity but beaten by NMI. The main reason can be that the purity measure simply summates all the majorities in clusters, while NMI will differentiate the duplicates.

Table 2. Comparison of clustering quality on CORA7

	CBKM	LKKM	IC	IC vs CBKM	IC vs LKKM
NMI	0.3752	0.4401	0.3908	+4.0%	-11.4%
Purity	0.5749	0.6329	0.6695	+16.3%	+5.7%

Table 3. Comparison of clustering quality on DBLP3

	CBKM	LKKM	IC	IC vs CBKM	IC vs LKKM
NMI	0.3774	0.5484	0.5110	+35.5%	-6.8%
Purity	0.7667	0.8607	0.8774	+14.3%	+2.0%

Table 4. Comparison of clustering quality on WIKI7

	CBKM	LKKM	IC	IC vs CBKM	IC vs LKKM
NMI	0.6467	0.6127	0.5959	-7.3%	-2.1%
Purity	0.8011	0.7711	0.7681	-4.1%	-0.4%

In Table 4, we find that, different from the above results, the performance of IC is worse than that of CBKM and LKKM on WIKI dataset. This can be arisen from the fact that the hyperlinks among web pages are not as reliable as the citation links and the co-authorship relations in scientific papers. Note that LKKM also performs worse than CBKM on this data set. Therefore, the usage of this hyperlink information has negative impacts on the clustering results.

In summary, compared with the iterative link based clustering algorithm LKKM, our IC approach has a small loss in performance. But, we also notice that the performance of IC is much better than that of the content based clustering method CBKM when the useful links like citations are explored. More importantly, IC gains a great improvement on efficiency over both the content and link based methods. As shown in Fig.3, the time cost for batch-mode clustering methods increases very fast as the size of collection size upgrades. In contrast, our IC algorithm is much less sensitive to it. This indicates that the proposed approach can apply to the large scale online environment.

6   Conclusion

In this paper, we exploit the problem of incremental clustering techniques for the dynamic document network, where the number of documents are growing continuously and the citation structure changes from time to time. We propose a novel approach to incorporate both the text and link information in each step of the incremental clustering procedure. Using this approach, the link impact is integrated to determine the host cluster of a document. Moreover, the core members in the network are used to represent the cluster and also filter out many check operations when updating the clustering distribution. Evaluation results show that our method achieves very significant improvements on efficiency and promising clustering qualities in comparison with other content and link based techniques. In the future, we plan to study the use of our method on more data sets with various types of links.

References

1. Angelova, R., Siersdorfer, S.: A neighborhood based approach for clustering of linked document collections. In: Proc. of the 15th ACM CIKM, pp. 778–779 (2006)
2. Angelova, R., Weikum, G.: Graph-based text classification: learn from your neighbors. In: Proc. of the 29th ACM SIGIR, pp. 485–492 (2006)
3. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Machine Learning 42, 143–175 (2001)
4. Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D.: A min-max cut algorithm for graph partitioning and data clustering. In: Proc. of the ICDM, pp. 107–114 (2001)
5. Ester, M., Kriegel, H.P., Sander, J., Wimmer, M., Xu, X.: Incremental clustering for mining in a data warehousing environment. In: Proc. of 24th VLDB, pp. 323–333 (1998)
6. Kazama, J., Torisawa, K.: Exploiting wikipedia as external knowledge for named entity recognition. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 698–707 (2007)
7. Li, H., Nie, Z., Lee, W., Giles, C., Wen, J.: Scalable community discovery on textual data with relations. In: Proc. of the 17th ACM CIKM, pp. 1203–1212 (2008)
8. Liu, X., Gong, Y., Xu, W., Zhu, S.: Document clustering with cluster refinement and model selection capabilities. In: Proc. of the 25th ACM SIGIR, pp. 590–599 (2002)
9. Menczer, F.: Lexical and semantic clustering by web links. JASIST 55, 1261–1269 (2004)
10. Nguyen-Hoang, T.-A., Hoang, K., Bui-Thi, D., Nguyen, A.-T.: Incremental Document Clustering Based on Graph Model. In: Huang, R., Yang, Q., Pei, J., Gama, J., Meng, X., Li, X. (eds.) ADMA 2009. LNCS, vol. 5678, pp. 569–576. Springer, Heidelberg (2009)
11. Ordonez, C., Omiecinski, E.: Frem: fast and robust em clustering for large data sets. In: Proc. the ACM CIKM, pp. 590–599 (2002)
12. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18, 613–620 (1975)
13. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. Tech. rep., University of Minnesota (2000)
14. Wang, J., Zeng, H., Chen, Z., Lu, H., Tao, L., Ma, W.Y.: Recom: Reinforcement clustering of multi-type interrelated data objects. In: Proc. of the 26th ACM SIGIR, pp. 274–281 (2003)
15. Zhang, X., Hu, X., Zhou, X.: A comparative evaluation of different link types on enhancing document clustering. In: Proc. of the 31st ACM SIGIR, pp. 555–562 (2008)
16. Zhao, Y., Karypis, G.: Criterion functions for document clustering: Experiments and analysis. Tech. rep., University of Minnesota (2002)
17. Zhao, Y., Karypis, G., Fayyad, U.: Hierarchical clustering algorithms for document datasets. Data Mining and Knowledge Discovery 10, 141–168 (2005)
18. Zhong, S.: Efficient online spherical k-means clustering. In: Proc. of IEEE IJCNN, pp. 3180–3185 (2005)
19. Zhou, X., Zhang, X., Hu, X.: Semantic smoothing of document models for agglomerative clustering. In: Proc. of the 20th IJCAI, pp. 2922–2927 (2007)

Integrating Temporal Usage Pattern into Personalized Tag Prediction

Lei Zhang, Jian Tang, and Ming Zhang

School of Electronics Engineering and Computer Science
Peking University, Beijing, China
{antiagainst,tangjianpku}@gmail.com, mzhang@net.pku.edu.cn

Abstract. The emergence of social tagging systems enables users to organize and share their interested resources. In order to ease the human-computer interaction with such systems, extensive researches have been done on how to recommend personalized tags for rescources. These researches mainly consider user profile, resource content, or the graph structure of users, resources and tags. Users' preferences towards different tags are usually regarded as invariable against time, neglecting the switch of users' short-term interests. In this paper, we examine the temporal factor in users' tagging behaviors by investigating the occurrence patterns of tags and then incorporate this into a novel method for ranking tags. To assess a tag for a user-resource pair, we first consider the user's general interest in it, then we calculate its recurrence probability based on the temporal usage pattern, and at last we consider its tag relevance to the content of the post. Experiments conducted on real datasets from Bibsonomy and Delicious demonstrate that our method outperforms other temporal models and state-of-the-art tag prediction methods.

1 Introduction

In recent years, social tagging systems, such like Delicious[1], Flickr[2] and Last.fm[3], have become more and more popular on the web. Unlike traditional ways of organizing resources using hierarchical categories, users of these systems can annotate resources with descriptive terms, called tags. Freely determined by users, tags are not only related to the content of resources but also to the specific users. Therefore, tags can be used to rediscover the corresponding web resources and to find users with similar interests. It is obvious that, as manual selected terms by human beings and concise summaries of resources, tags can also be utilized in other areas such as web search [1] [3] and personalized search [10] [2] [7]. However, since only a very small proportion of web resources are saved in social tagging systems, fully incorporating tagging information into such area is restricted. So, automatic tagging is of crucial importance. Besides, users prefer to

---

[1] http://www.delicious.com
[2] http://www.flickr.com
[3] http://www.last.fm

select among candidates of tags instead of finding the right ones by themselves since the former is much easier and offers better human-computer interaction experiences than the latter. This also requires methods for recommending tags to be developed.

Extracting from resources' textual information (e.g., title, URL, web page) directly is the most natural way for finding tags to recommend. It is fail-safe, but cannot provide categorical and synoptic terms that are absent. Collaborative filtering, a widely used method in other recommendation systems, can be used to recommend tags but has limitations too. The main idea of this method is to find similar users, usually a k-neighborhood, and then use these like-minded users' profiles to calculate a prediction. However, similarity between users is not always that reliable. Using resources as objects suffers from the very low recurrence rate of resources. In a real world scenario, the probability of a resource being new to tag recommenders is more than 0.9 [13]. On the other side, using tags as objects relies only on some very general and widely used terms. Combining these methods can get better performance, but not fix the problem of neglecting the importance of personalization and modeling users' behaviors more precisely.

One possible solution is exploiting the graph structure of social tagging systems, called graph-based methods. These methods achieve higher accuracy by deploying weight-spreading or matrix factorization schemes on the connection among users, resources and tags. The targeted user's connection is favored in this process, thus more personalized. However, graph-based methods are highly calculation intensive, so only the dense part of the data, such as the remainder after a CORE p [5] process, can be used in practice, which is inconsistent with the reality. Another way is called content-based methods, infering users' preferences from all textual related information (e.g., tags used, titles and contents of resources). Although not accurate as graph-based methods, these methods have the advantage that it can predict tags for any user and any web resource.

Apart from graph structure and textual information, recently researchers begin to notice the importance of temporal factor in modeling user behavior. Zhang et al. [14] analyse the recurrence dynamics of social tagging, showing that more frequently and recently used tags should be emphasized when recommending. Another work [11] reveals users' session-like behaviors in social tagging systems and demonstrates that the predictive performance can be improved by considering sessions significantly. Inspired by these findings, we also focus on time factor. Our method regards the ranking of a tag as a measure of the tag's consistency with the user's general interests, short-term interests and the resource content. User's general interests are represented as a multinomial distribution on tags. In order to find user's short-term interests, we investigate the first- and last-time usage pattern of users' tags. Combining these with the relevance of tag and resource, we propose a novel approach for the tag prediction problem.

The rest of the paper is organized as follows: Section 2 reviews some recent work on personalized tag recommendation. Section 3 formally formulates the problem and describes our proposed method in detail. Then we present and

evaluate our experimental results on two real-world datasets in Section 4 and conclude our work in Section 5.

2   Related Work

Recently, more and more researchers begin to focus on user-specific information. They use this kind of information to construct profiles representing users' preferences more precisely. Generally speaking, there are two main directions of personalized tag recommendation, content-based and graph-based. Content-based methods tend to extract useful information from all the resources and tags used by the targeted user. Such methods work well when faced with new users or new resources. Lipczak et al. [6] discuss the potential role of three tag sources: resource content as well as resource and user profiles. Their system compiles a set of resource specific tags, which includes tags related to the title and tags previously used to describe the same resource (resource profile). These tags are checked against user profile tags – a rich, but imprecise source of information about user interests. The result is a set of tags related both to the resource and user. Yin et al. [13] propose a probabilistic model following a Bayesian approach, and integrate three factors – ego-centric effect, environmental effect and web page content. As for graph-based methods, FolkRank [4], inspired by PageRank, is of great importance. Like PageRank, it employs the random surfer model and spreads the weight through the tripartite graph of users, resources and tags recursively until convergence. Factorization models are also used in this area and show very good performance [8] [9].

However, the above researches treat users' preference of tags as constant over time. Recently researchers begin to realize the importance of time in constructing user profile. Zhang et al. [14] study the recurrence dynamics of tags, and find more frequently or recently used tags should be favored for tag suggestion, because they have higher probability to be reused. Yin et al. [11] reveal session-like behavior in social tagging systems and propose a session-based method. They divide users' tagging history into sessions by comparing the Jaccard's coefficient of two consecutive posts to a threshold. Recommendation bifurcates according to whether the post, whose tags are to be recommended, is a topic switch post. If it is not, tags are recommended from the former session, otherwise, tags are recommend using an external method [6]. And the probability of current post being the start of a new session is learned from time intervals in the past. The method is simple but can outperform two state-of-the-art algorithms significantly. Yin et al. [12] propose a user-tag-specific temporal interest model for tracking user interests over time. They believe once an interest appears, it will stay for a while and then become weaker and weaker. The speed of decay is different for different users, even for different tags of the same user. A user's current interest in a tag is the accumulation of the residual interests (i.e., intersts that the user still has in the tag after decay) of all its former occurrences.

3   Tag Recommendation

In order to be accurate, tags predicted should not only relevant to the resource, but also reflect the user's interests. Users' general interests are not very easy to capture, and users' short-term interests change from time to time.

3.1   Definitions

A social tagging system consists of users $u \in U$, resources $r \in R$, tags $t \in T$ and timestamps $\tau \in M$. An annotation is the relation among the four types of objects, $a \in A \subseteq U \times R \times T \times M$. We call all annotations to a resource by a user a post $p \in P = \{(u, r, \tau) \mid \exists t \in T : (u, r, t, \tau) \in A\}$.

So, the tag recommendation problem is, when the current user u_0 is trying to save a resource r_0 at time τ_0, to provide a ranked list of tags based on the user's past posts $S = \{(u, r, \tau) \mid \exists t \in T : (u, r, t, \tau) \in A, u = u_0, \tau < \tau_0\}$.

3.2   Tag Relevance to Resource

Social tagging systems enable users to organize and share resources of different types, such like Delicious for web pages, Bibsonomy[4] for papers, Last.fm for music and Flickr for pictures. Web pages and papers have rich textual information, thus it is relatively easy for us to extract synoptical terms to describe these materials. When it comes to music and pictures, descriptive terms can hardly be derived from their "contents" automatically. Although many researches have been done in order to tackle this problem, there seems no major breakthrough. Fortunately, these resources usually have titles and universal resource identifiers which can help us grasp some idea about their topics. So, in order to be general, we use titles and URLs of the resources to represent their contents.

Following the notations and reasoning used by Yin et al. [13], we denote $P(r|u, t)$ as the occurrence probability of resource r in the set of resources which are tagged by t. The content of resource r can be treated as a bag of words $W = \{w|w \text{ appears in resource } i\}$. So, using the unigram language model, we can rewrite $P(r|u, t)$ as

$$P(r|u, t) = \prod_{w \in W_r} P(w|u, t) \tag{1}$$

Equation 1 breaks the resource-level probability $P(r|u, t)$ into the production of word-level probabilities. $P(w|u, t)$ means the likelihood of the word w appearing in the contents of resources tagged by t. Since the discussion is for a particular user u, for the sake of simplicity, we just leave out u in the following equations.

Given x (some tag or resource), denote the number of occurrence of y (some resource or word) as $N_{y,x}$. Then we have the following:

$$N_{w,t} = \sum_{r \in R} N_{w,r} \cdot N_{r,t} \tag{2}$$

---

[4] http://www.bibsonomy.org

To estimate $P(w|t)$, we assume words obey the following distribution:

$$P(w|t) = \frac{N_{w,t}}{N_t} \tag{3}$$

Then, according to maximum likelihood estimation (MLE), the probability of word w with respect to tag t can be maximized when

$$N_t = \sum_w N_{w,t} \tag{4}$$

So, $P(w|t)$ is calculated as

$$P(w|t) = \frac{N_{w,t}}{\sum_w N_{w,t}} \tag{5}$$

3.3   User's General Interests

Users' true and complete general interests are always hidden. Even a user himself/herself cannot point out clearly a ranked list of topics that attract him/her. Since resources are deliberately saved and tags are manually selected by users, a user's tag history, i.e., all the tags used by him/her, reveals partially the user's general interests. Although not very precise, it is a feasible indicator.

We assume user's general interests follow a multinomial distribution. According to MLE, we can represent a user's preference to a tag t as following:

$$P(t|u) = \frac{N_{t,u}}{\sum_t N_{t,u}} \tag{6}$$

So, $P(t|u)$ is calculated as tag t's occurrence count out of the size of user u's tag vocabulary.

3.4   Temporal Tag Usage Patterns

In the long-term, user's general interests may be relatively stable. But in the short-term, a user usually focuses on some very specific topics. In social tagging systems, this phenomenon is more obvious. And there are relations between short-term interests, i.e., relations like the first giving rise to the second, which then switches to the third. The following is an intuitive explanation. When surfing on the internet, a user may run into a resource of his/her interest. If the resource covers some information unfamiliar to the user, he/she may adds the resource into the social tagging system and then digs more out about the information. So, many other resources related to the information are likely to be saved consecutively. In this process, other new information may also attracts the

user and triggers a sequence of resources to be saved. Once acquainted with the information, the user may transfer to other information that he/she is unfamiliar with. The next time this user meets something new, he/she may do the same routine.

To investigate the temporal dynamics of user behavior stated above is not an effortless undertaking. We focus on the first- and last-time usage of tags under the consideration that, the first-time usage of a tag indicates that the user begins to notice this topic and the last-time usage indicates the last time when the user give notice to this topic.

Formally, for a user u, denote $S_u = (p_1, p_2, \cdots, p_i, \cdots, p_n)$ as the sequence of u's posts in chronological order so far. Their timestamps $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_i \leq \cdots \leq \tau_n$. Let $S_{u,t} = (p_{i_1}, p_{i_2}, \cdots, p_{i_m})$ be the subsequence of S_u where tag t appears. Then we define the distance to the first- and last-time usage as

$$d^f_{u,t,p_{i_k}} = \log(i_k - i_1) + c^f, \qquad p_{i_k}, p_{i_1} \in S_{u,t} \tag{7}$$

$$d^l_{u,t,p_{i_k}} = \log(i_k - i_{k-1}) + c^l, \qquad p_{i_k}, p_{i_{k-1}} \in S_{u,t} \tag{8}$$

where c^f and c^l are two constants.

That is, given u, t, p_i, the distance to first-time usage is just the logarithmic index difference between p_i and the first post in S_u tagged by t. Logarithm is used to scale the difference. We count the distance for every (u, t, p_i) tuple, and show the result in Figure 1. The Figure shows that the earlier a tag is first used, the less likely it is to recur again. This can be explained in a way that is consonant with the above explanation. Earlier tags represent information unfamiliar to the user in earlier time. The lager the distance to the first-time usage, the more time the user have to get acquainted with the information and the less likely it is interesting to the user now.

As for distance to last-time usage, it is the logarithmic index difference between p_i and the former post in S_u tagged by t. We also count the distance for every (u, t, p_i) tuple, and show the result in Figure 1. The y-axis is logarithmically scaled. As expected, the more recently a tag is used, the more likely it is to be reused. This supports our idea about relations between short-term interests. Recently used tags represent topics that users are interested in at the current moment or topics that spawn the current topic.

Based on the observations above, we propose a novel method for measuring the recurrence probability of a tag, merely from the perspective of the tag's first- and last-time usage patterns. Mathematically, recurrence likelihood of a tag is expressed as

$$P^r_{u,t} = d^f_{u,t,p} \cdot (d^l_{u,t,p})^{-d^f_{u,t,p}} \tag{9}$$

That is, recurrence likelihood of a tag should decrease since its first-time usage, i.e., the first time it attracts the user. We use the distance to last-time usage as the base to tune the recurrence likelihood of the tag in latter posts.

Yin et al. [12] also use exponential function to describe the residual interests of tags, but their model assumes tag occurrences are independent. So, if a tag occurs many times, their influences will accumulate instead of just using the last
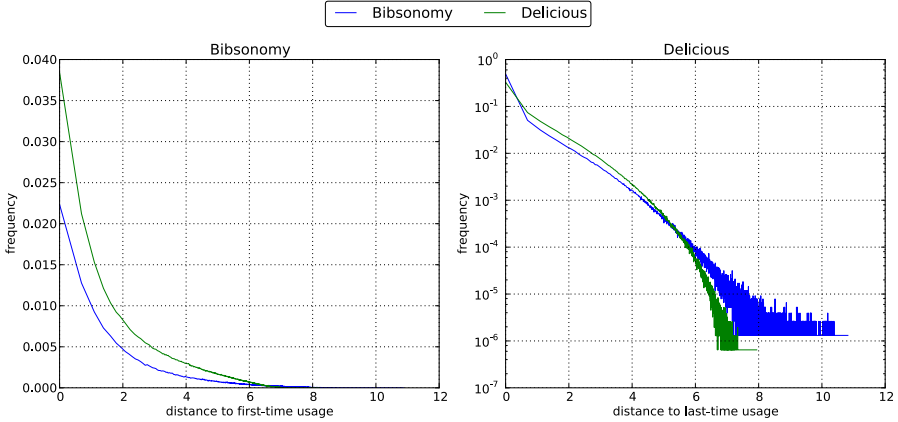
Fig. 1. Distribution of tag's distance to first- and last-time usage

one. Besides, their method is based on the real time intervals while ours relies on relative positions. To our opinion, real time intervals are less reliable than relative positions because real time intervals are influenced by many uncontrollable factors and can vary wildly. For example, a user may save one resource at the night and then save another similar resource the next morning, such resulting a large time gap. Or, a user can save two related resources one by one very quickly. Both are resources with the same topic, but time intervals differ greatly. On the contrary, relative positions do not have such problems.

With the above three parts measuring tag relevance to the resource, user's general interests and user's short-term interests respectively, we can rank tags using the following scheme:

$$R_{u,t} = P(r|u,t) \times P(t|u) \times P_{u,t}^r \tag{10}$$

Since it is possible that the user for whom we want to predict tags have no tagging behaviors before, we also need a way to solve this special issue. It is obvious that, under such circumstances, no personalized technique can be used. So, when confronted with such user, we use the content-based method LHKM [6] instead.

4   Experiments

In this section, we firstly describe our datasets and evaluation methodology. Then we examine the three components of our method. At last we compare our method with existing temporal models and two state-of-the-art methods.

4.1   Datasets

Our experiments are conducted on two datasets: Delicious and Bibsonomy. Delicious is a popular bookmarking system in which users annotate URLs. We

collected the complete profiles of thousands of users, and then randomly select 4,614 users for our experiments. Bibsonomy enables its users to annotate both URL bookmarks and journal articles. Our dataset of Bibsnomy, provided by ECML PKDD Challenge 2009, is an almost complete dump of the Bibsonomy website before January 1st, 2009. The basic statistics are shown in Table 1.

We use the evaluation method which is proposed in [13] and much closer to the real world. This method, called online framework, samples test posts randomly from the whole dataset along the timeline. For users and resources in the test dataset, posts having earlier timestamps than the test posts are used for training. So, the training data is different for every test post. We randomly sample 1,849 posts from Delicious and 676 posts from Bibsonomy as test posts.

Table 1. Datasets

	#users	#resources	#tags	#posts	#annotations
Bibsonomy	2,679	235,328	56,424	263,004	916,469
Delicious	4,614	603,151	130,909	714,597	2,151,678

4.2   Evaluation Methodology

The performance of recommendation is evaluated on their ability to recommend tags given a user-resource pair (u, r). The user-resource pair is submitted to the recommenders and then the set of tags returned $T_{u,r}^{pred}$ are compared against the real tags $T_{u,r}^{real}$.

Precision and recall are common metrics for evaluating the utility of recommendation algorithms. Precision measures the exactness of the recommendation results while recall measures the completeness of the recommendation results. Given the test set T_{test},

$$precision = \frac{1}{|T_{test}|} \sum_{(u,r) \in T_{test}} \frac{|T_{u,r}^{pred} \cap T_{u,r}^{real}|}{|T_{u,r}^{pred}|} \tag{11}$$

$$recall = \frac{1}{|T_{test}|} \sum_{(u,r) \in T_{test}} \frac{|T_{u,r}^{pred} \cap T_{u,r}^{real}|}{|T_{u,r}^{real}|} \tag{12}$$

Precision and recall will vary with the size of the recommendation set. By adjusting the recommendation set size from 1 to 10, we will get 10 datapoints on precision-recall plot. A curve is drawn by connecting the 10 datapoints and then used to show the performance of the recommendation method.

4.3   Evaluating the Functionalities of Different Parts

Our method for ranking a tag is mainly comprised of three components: user's general interest in it, its recurrence probability based on temporal usage pattern

and its relevance to the resource. We separate the three parts and then evaluate their functionalities. Results are shown in Figure 2, where "GI", "RP" and "TM" represent the three part mentioned above respectively. Besides, LHKM [6] is used as the fourth part to handle new users. c^f in Equation 7 and c^l in Equation 8 are constants to avoid zero distances. They should be small; otherwise, the first-and last-time usage will not have much influence over the recurrence probability. After trying various combinations, we set c^f as 2 and c^l as 1. A datapoint on a curve stands for the size of the recommendation set, starting with one tag on the upper left of the curve and ending with ten tags on the lower right.
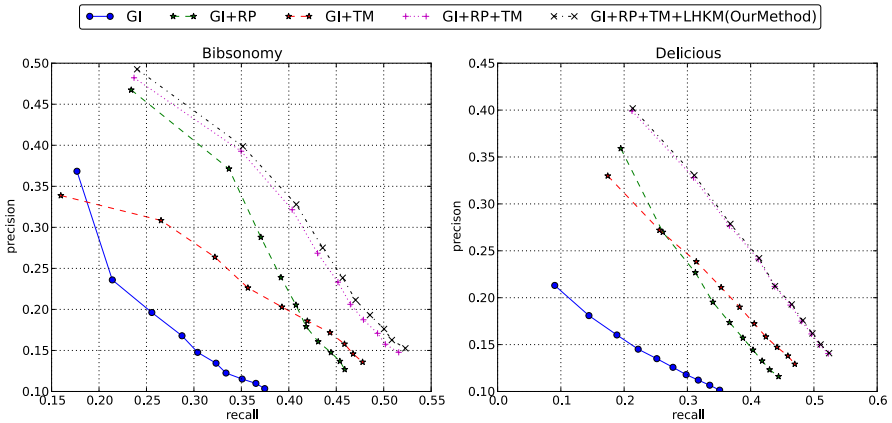
Fig. 2. Evaluating the functionalities of different parts

User's general interests itself achieve neither high precision nor high recall because it simply recommends the most frequent tags used in the past. With tag relevance to resources added, recall is enhanced significantly since tags related to the resource tend to have a higher tag-relevance value. The third part, recurrence probability from temporal tag usage pattern, improves precision greatly because it emphasizes the tags that the user are interested currently and want to use again. By combining the three part together, our method works well both in precision and recall.

In our test set, 44 out of 676 (about 6.5%) in Bibsonomy and 9 out of 1,849 (about 0.5%) in Delicious are new users. Under such circumstances, no person-alized technique can be used. So, when confronted with such user, we use the content-based method LHKM [6] instead. Since only a small percentage of users are totally new to the system, the result is slightly improved.

4.4   Comparison with Other Methods

Most of the researches on personalized tag prediction assume that users' interests do not change with time. We call this the long-term model, which is just the

general interests part used in our method. Two recently proposed method for modeling the temporal dynamics of users' interests are the session-based method [11] and user-tag-specific temporal interest model [12]. User-tag-specific temporal interest model, abbreviated as user-tag TIM, uses the same method with session-based method to handle topic switch. Since Delicious do not have detailed time information of the posts which are required to detect topic switch, we only compare our methods with the above three temporal models. Results are shown in Figure 3. The figure shows that session-based method and user-tag TIM are somewhat evenly matched and our method outperforms both of them. Session-based method does better than user-tag TIM when the recommendation set size is small, but its performance almost remains the same when the set size becomes sufficiently bigger. Our methods can predict far more accurate and complete tags
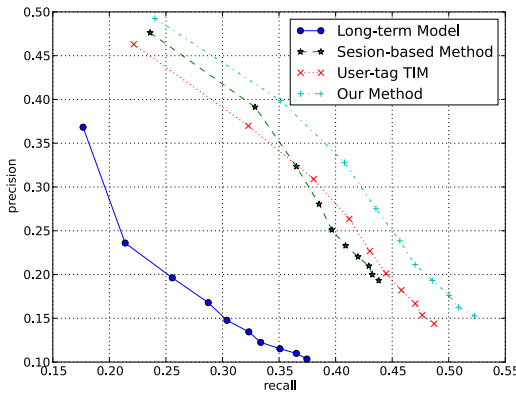

Fig. 3. Comparison with other temporal models on Bibsonomy (Comparison is not conducted on Delicious because Delicious lacks detailed time information)
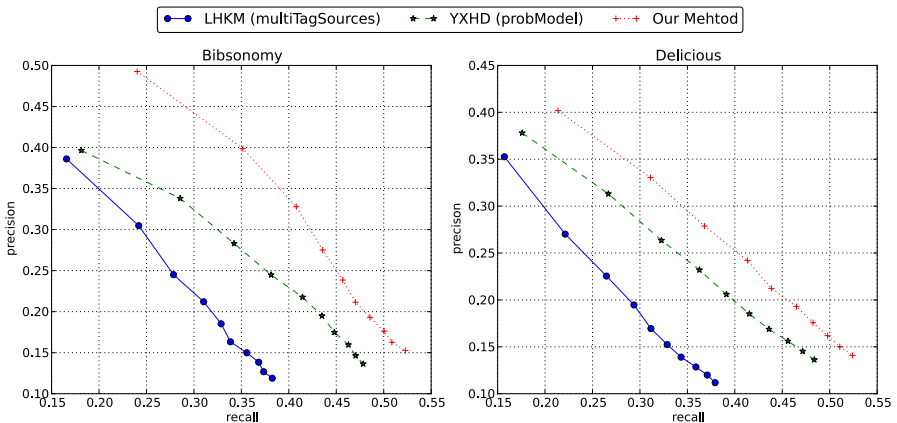

Fig. 4. Comparison with state-of-the-art methods

than user-tag TIM no matter what the size of the recommendation set is. When the size of recommendation set is sufficiently large, our method outperforms session-based method significantly in recall.

We also compare our method with two state-of-the-art content-based method, LHKM [6] and YXHD [13]. LHKM took the first place in the "content-based" recommendation task of the ECML PKDD Discovery Challenge 2009. YXHD uses a probabilistic model and treats the tag prediction problem as the reverse problem of web searching. From Figure 4 we can see that our method perform better than both of them significantly.

5   Conclusion

In this paper, we have investigated temporal tag usage patterns, especially the first- and last-time usage. We find that the oldness of tags have negative influence on their recurrence probability while the recency of tags have positive influence. By combining these observations with the user's general interests and tag relevance to the resource, we propose a new method for recommending tags. Experiments conducted on the datasets of Bibsonomy and Delicious show that our method outperform the session-based method, the user-tag-specific temporal interest model, and two other state-of-the-art content-based methods.

References

[1] Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, pp. 501–510. ACM, New York (2007)

[2] Cai, Y., Li, Q.: Personalized search by tag-based user profile and resource profile in collaborative tagging systems. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 969–978. ACM, New York (2010)

[3] Heymann, P., Koutrika, G., Garcia-Molina, H.: Can social bookmarking improve web search? In: Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, pp. 195–206. ACM, New York (2008)

[4] Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information Retrieval in Folksonomies: Search and Ranking. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)

[5] Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag Recommendations in Folksonomies. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 506–514. Springer, Heidelberg (2007)

[6] Lipczak, M., Hu, Y., Kollet, Y., Milios, E.: Tag sources for recommendation in collaborative tagging systems. In: Eisterlehner, F., Hotho, A., Jäschke, R. (eds.) ECML PKDD Discovery Challenge 2009 (DC 2009). CEUR-WS.org, vol. 497, pp. 157–172 (September 2009)

[7] Noll, M.G., Meinel, C.: Web Search Personalization Via Social Bookmarking and Tagging. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 367–380. Springer, Heidelberg (2007)

[8] Rendle, S., Marinho, L.B., Nanopoulos, A., Schmidt-Thieme, L.: Learning optimal ranking with tensor factorization for tag recommendation. In: The 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009), pp. 727–736. ACM, New York (2009)

[9] Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Davison, B.D., Suel, T., Craswell, N., Liu, B. (eds.) WSDM, pp. 81–90. ACM (2010)

[10] Xu, S., Bao, S., Fei, B., Su, Z., Yu, Y.: Exploring folksonomy for personalized search. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, pp. 155–162. ACM, New York (2008)

[11] Yin, D., Hong, L., Davison, B.D.: Exploiting session-like behaviors in tag prediction. In: Proceedings of the 20th International Conference Companion on World Wide Web, WWW 2011, pp. 167–168. ACM, New York (2011)

[12] Yin, D., Hong, L., Xue, Z., Davison, B.D.: Temporal dynamics of user interests in tagging systems. In: Burgard, W., Roth, D. (eds.) AAAI. AAAI Press (2011)

[13] Yin, D., Xue, Z., Hong, L., Davison, B.D.: A probabilistic model for personalized tag prediction. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2010, pp. 959–968. ACM, New York (2010)

[14] Zhang, D., Mao, R., Li, W.: The recurrence dynamics of social tagging. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009, pp. 1205–1206. ACM, New York (2009)

An Optimization Strategy for Mashups Performance Based on Relational Algebra

Hailun Lin[1,2], Cheng Zhang[1], and Peng Zhang[1,2]

[1] Institute of Computing Technology, Chinese Academy of Sciences, 100190, Beijing, China
[2] Graduate University of Chinese Academy of Sciences, 100049, Beijing, China
{linlunnian,zhch,zhangpeng}@software.ict.ac.cn

Abstract. Recently, mashups as a new type of application have gained tremendous popularity, which provide opportunities of creating personalized Web applications using Internet-based resources to end-users. Meanwhile, the performance of mashups can not be neglected while the end-users participate in mashups construction. Nowadays, there is a lot of research work with a focus on developing tools or platforms to support mashups construction. However, there are few studies concerned about the performance of mashups. In order to improve mashups performance, this paper draws on experience of relational algebra query optimization to establish mashup query-tree model and mashup operator performance model, and defines mashup operators' equivalent transformation rules and query-tree heuristic rules. On this basis, this paper proposes an optimization algorithm for mashups performance. Experiments show that our strategy can effectively improve the run-time performance of mashups.

Keywords: Web2.0, mashup, relational algebra, query optimization.

1   Introduction

In recent years, mashups have gained tremendous popularity, which have become an important part of Web2.0 applications. Mashups are web applications generated by combining content, presentation, or application functionality from disparate web sources [1]. Nowadays, some companies and research institutions have developed many mashups construction tools, for example, Pipes [2], Damia [3], Marmite [4], SpreadMash [5], Mashroom [6]. The common goal of these tools is to hide the complex processing and programming aspects of building web-based applications for non-technical end-users and enable end-users steer the construction of personalized applications. In fact, the process of mashups construction actually expresses the end-users' demand on data sources, and the operators and their sequence are up to users rather than the system. However, the sequence usually has a great impact on mashups performance. For instance, one mashup first filters "comedy" video list from video.google.com according to the hot video category, and then sorts the list by time, another mashup sorts these video ahead of filtering. The former one's performance is better than the latter one even if they have the same results. This is because filtering

generally makes the intermediate results for computation much smaller. Therefore, the sequence of operators selected by end-users directly affects the run-time performance of mashups.

It can be seen by the analysis that mashups performance is a noticeable challenge. In order to improve mashups performance, this paper presents an optimization strategy based on relational algebra, which employs operators' equivalent transformation rules and query-tree heuristic rules [7] to enhance the efficiency of mashups by reordering mashups operators, of which has the following special features.

1) We establish mashup query-tree model and operators' performance model.
2) Based on operators' performance model, we present operator equivalent transformation rules and mashup query-tree heuristic rules, and propose a mashups performance optimization algorithm.

The rest of this paper is organized as follows: Section 2 reviews related works. Section 3 introduces mashup query-tree model and operator performance model. Section 4 introduces mashups performance optimization. Section 5 is experiment and discussion. Section 6 concludes the paper.

2   Related Works

In recent years, mashups technology has been widely developed, there have been many mashups tools and platforms boomed to support just-in-time mashups construction. Based on adopted programming model, Wang [6] divided these tools into four categories: 1) flow-chart-like programming tools, such as Pipes [2]; 2) spreadsheet-like programming tools, typical representatives are SpreadMash [5], EditGrid [8]; 3) tree-based programming tools, MashMaker [9] falls into this category; 4) browser-centric programming tools, d.mix [10] belongs to this category. Lorenzo [11] has analyzed the advantages and disadvantages of existing mashups tools, but there are seldom works focusing on mashups performance.

In order to improve the efficiency and run-time performance of mashups, one way is by caching. Hassan has presented a dynamic caching framework MACE [12] which employs caching mechanism. In order for MACE to select stages at which data will be cached, it continuously observes the execution of mashups, and collects statistics such as request frequencies, update rates and cost and output size values at various nodes of mashups. It then performs cost-benefit analysis of caching at different nodes of mashups, and chooses a set of nodes that are estimated to yield best benefit-cost ratios. For each new mashup, MACE platform analyzes whether any of the cached results can be substituted for part of the mashup workflow. If so, the mashup is modified so that cached data is re-used. To a great extent, MACE enhances the scalability and performance of mashups. This method has remarkable effect on mostly static resources, but does not adapt to resources change frequently. This is because it will continuously maintain consistency of cached data.

In addition, the literature [13] provides AMMORE platform to support mashups construction. AMMORE represents mashups as strings, it detects the longest common

components across mashups, and merges mashups with other mashups by common components to minimize the total number of operators a mashup platform has to execute. The results of longest common components can be used by multiple mashups. This way enhances the scalability and performance of mashup platforms, but the complexity and cost of calculating the longest common components across mashups also increases with the increase of mashups hosted by the platform. Moreover, AMMORE defines a set of canonical forms for mashups operators, which improves mashups efficiency by reordering mashups operators based on these canonical forms. We draw lessons from this work for our mashups performance optimization.

3   Mashup Query-Tree and Operator Performance Model

Generally, mashups platform has a basic operators set. The operators set discussed in this paper includes *count*, *filter*, *union*, *unique*, *join*, *sort*, *projection*, *truncate*, *tail*, *rename*, which are analogous to the operators in the relational algebra targeting at relational data. In addition, it also includes two special operators: *fetch* and *dispatch*, where *fetch* is used to access data from web-based resources and wraps obtained data with relational data model [14]; *dispatch* is used to provide mashups results to end-users. We use *opset* to represent the operators set, *opset*={*fetch*, *count*, *filter*, *union*, *unique* ,*join*, *sort*, *projection*, *truncate*, *tail*, *rename*, *dispatch*}. These operators are summarized from several mashups platforms [2-6], therefore we will not repeat the specific definitions of these operators, but focus on their composition sequences.

From this perspective, we divide these operators into three categories: data access operators, data query operators and data present operators. Data access operators wrap data resources and provide data access interfaces, here we use *fetch* operator to represent them. Data query operators include *count*, *filter*, *union*, *unique*, *join*, *sort*, *projection*, *truncate*, *tail*, *rename*, these operators are used to transform data fetched by *fetch* operator to meet end-users' requirements. Data present operators are used to ship mashups results to end-users, here we use *dispatch* operator to represent them.

3.1   Mashup Query-Tree

From the above analysis, each mashup (or short M) should include one *fetch* operator at least, one *dispatch* operator and a few of data query operators, we use a set named *mOpset* to express operators used in a mashup, which are chosen from *opset*. A mashup can be written as equation (1), where Q refers to data query operators.

$$M = \underline{fetch}^+ + Q^* + \underline{dispatch} \tag{1}$$

Combining with tree representation of relational algebra expressions, a mashup can also be expressed with a tree structure with each node corresponding to an operator: The root node refers to a data present operator; the leaf nodes represent data access operators; other nodes represent data query operators. Then, we use query-tree model to

represent a real mashup. We take a quite popular application from Yahoo! Pipes community[1] named Aggregated News Alerts[2], which setups a persistent search at Bloglines, Findory, Google Blog Search, Google News, IceRocket, MSFT Live News, Technorati and Yahoo! News. Its query-tree can be illustrated in Fig. 1.

Each operator in a mashup has an execution cost, hence the execution cost of a mashup is the sum execution cost of operators of the mashup. This paper takes the reciprocal value of $m(t)$ which is the sum execution cost of the mashup as mashup efficiency, the smaller of $m(t)$, the higher of the mashup efficiency. We test the execution cost of the non-optimized Aggregated News Alert, the average cost of 100 request times is 1658.5ms. There would be higher cost when the fetched data is large-scale. In order to implement mashups performance optimization, we define operator performance model to optimize the sequence of these operators.
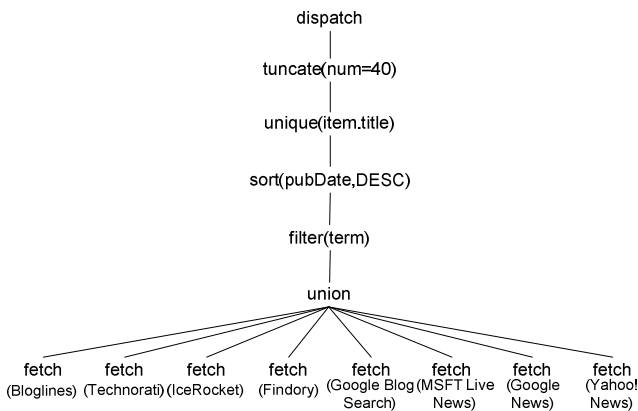

Fig. 1. The query-tree of Aggregated News Alerts

3.2   Operator Performance Model

We employ the work of operator scheduling in data stream systems [15,16], which is mainly carried out for data query operations, to define mashups operator performance model. Before introducing operator performance model, we first give two definitions.

Definition 1. Given data set D and operator p, p is performed on D, the cost is t, if D contains n tuples, the average time of processing each tuple is t/n, which is called the execution cost of p, denoted by $c(p)$.

Definition 2. Given data set D and operator p, p is performed on D, the output result set is R, if D contains n tuples, R contains m tuples, we call m/n is the selectivity of p, denoted by $s(p)$.

---

[1] http://pipes.yahoo.com/pipes/pipes.popular.
[2] http://pipes.yahoo.com/pipes/pipe.info?_id=fELaGmGz2xGtBTC3qe5lkA.

Based on operator execution cost $c(p)$ and selectivity $s(p)$, we define operator performance model in order to optimize the sequence of two operators.

Definition 3. Given two operators p_i and p_j, with p_j following p_i. In this case, we can compute the total cost of each input tuple for p_i and p_j as $c(p_i)+s(p_i)\times c(p_j)$; Reversing the operators gives a like computation, the sum cost is $c(p_j)+s(p_j)\times c(p_i)$. If $(c(p_i)+s(p_i)\times c(p_j))>(c(p_j)+s(p_j)\times c(p_i))$, it reveals that reversing the two operators can enhance the efficiency of mashups. Therefore, to determine whether two operators should switch we give the following condition:

$$(1-s(p_j))/c(p_j)>(1-s(p_i))/c(p_i) \tag{2}$$

If p_i and p_j satisfy formula (2), then exchange the order of p_i and p_j; else stay in the same order. We call $(1-s(p))/c(p)$ as the performance model of operator p, denoted by $m(p)$. Then:

$$m(p)=(1-s(p))/c(p) \tag{3}$$

Computing each operator in a mashup according to formula (3), it is easy to obtain an optimal order by sorting all operators according to their corresponding results in decreasing order. However, only optimizing mashup based on operator performance model is not enough, meantime it should consider the semantic association between mashup operators. To illustrate this case, we take Aggregated News Alerts for example. According to operator performance model, the *sort* operator should follow the *unique* and *truncate* operators, in this case, the application's efficiency will be improved, but the obtained results are not consistent with end-user's demand. In this respect, this paper proposes mashups performance optimization algorithm combining with operators' semantic association. Following, we first discuss mashup operators' equivalent transformation rules, and then define mashup query-tree heuristic rules, and last present mashup performance optimization algorithm.

4   Optimization for Mashups Performance

4.1   Operator Equivalent Transformation Rules

To simply the writing, here using Σ, σ, $\cup$, μ, $\rhd_j$, s, π, t, τ, r respectively represents *count*, *filter*, *union*, *unique*, *join*, *sort*, *projection*, *truncate*, *tail*, *rename*. By analogy with relational algebra expression equivalent transformation rules, mashup operator equivalent transformation rules mainly consist of commutative law, distributive law and concatenate law. This paper uses $R(D)$ and $Tuple(D)$ to represent the schema of D and the returned results set of calling D. Given two data sources D_1 and D_2, and two operators p_1 and p_2, p_1 and p_2 have the following rules as shown in Table 1, where F is filter condition, A_i ($i=1,2,\ldots,n$) and B_j ($j=1,2,\ldots,m$) are attributes of data source, d_i ($i=1,2,\ldots,n$) is sort style and $d_i \in \{ASC,DESC\}$; $<A_i,d_i>$ expresses sorting data source on A_i in manner d_i.

Table 1. The operator equivalent transformation rules

Category	Rule	Description
Commutative Laws	$\sigma_F(\mu_{A1,A2,\ldots,An}(D_1))\equiv\mu_{A1,A2,\ldots,An}(\sigma_F(D_1))$	law of filter and unique
	$\sigma_F(s_{<A1,d1>,<A2,d2>,\ldots,<An,dn>}(D_1))\equiv s_{<A1,d1>,<A2,d2>,\ldots,<An,dn>}(\sigma_F(D_1))$	law of filter and sort
	$\pi_{A1,A2,\ldots,An}(\sigma_F(D_1))\equiv\pi_{A1,A2,\ldots,An}(\sigma_F(\pi_{A1,A2,\ldots,An,B1,B2,\ldots,Bm}(D_1)))$	law of filter and projection
	$\mu_{B1,B2,\ldots,Bm}(s_{<A1,d1>,<A2,d2>,\ldots,<An,dn>}(D_1))\equiv s_{<A1,d1>,<A2,d2>,\ldots,<An,dn>}(\mu_{B1,B2,\ldots,Bm}(D_1))$	law of unique and sort
Distributive Laws	$\pi_{A1,A2,\ldots,An}(D_1\cup D_2)\equiv\pi_{A1,A2,\ldots,An}(D_1)\cup\pi_{A1,A2,\ldots,An}(D_2)$	law of projection and union
	$\sigma_F(D_1\cup D_2)\equiv\sigma_F(D_1)\cup\sigma_F(D_2)$	law of filter and union
	$\sigma_F(D_1\triangleright_jD_2)\equiv\sigma_F(D_1)\triangleright_j\sigma_F(D_2)$	law of filter and join
	$\pi_{A1,A2,\ldots,An}(D_1\triangleright_jD_2)\equiv\pi_{A1,A2,\ldots,An}(D_1)\triangleright_j\pi_{A1,A2,\ldots,An}(D_2)$	law of projection and join
Concatenate Laws	$\pi_{A1,A2,\ldots,An}(\pi_{B1,B2,\ldots,Bm}(D_1))\equiv\pi_{A1,A2,\ldots,An}(D_1)$	law of projection
	$\sigma_{F1}(\sigma_{F2}(D_1))\equiv\sigma_{F1\wedge F2}(D_1)$	law of join

These rules can be proved by the set operations. Following, we prove the distribute law between *filter* and *union* to illustrate the rationality and validity of the rules:

Rule. The distributive law between *filter* and *union*

$$\sigma_F(D_1\cup D_2)\equiv\sigma_F(D_1)\cup\sigma_F(D_2)$$

Proof: Assuming the schema of D_1 and D_2 is A, denoted by $R(D_1)=R(D_2)=A$. $R(\sigma_F(D_1\cup D_2))=R(\sigma_F(D_1)\cup\sigma_F(D_2))=A$; Moreover, according to the properties of collection operations, $\forall t\in Tuple(\sigma_F(D_1\cup D_2))$, then $t\in Tuple(\sigma_F(D_1))$ or $t\in Tuple(\sigma_F(D_2))$, it can be concluded $t\in (Tuple(\sigma_F(D_1))\cup Tuple(\sigma_F(D_2)))$, hence $Tuple(\sigma_F(D_1\cup D_2))\subseteq(Tuple(\sigma_F(D_1))\cup Tuple(\sigma_F(D_2)))$; similarly available, $(Tuple(\sigma_F(D_1))\cup Tuple(\sigma_F(D_2)))\subseteq Tuple(\sigma_F(D_1\cup D_2))$. Therefore, *filter* and *union* satisfy distributive law.

Other rules can be proven by referring to this proof.

4.2 Mashup Query-Tree Heuristic Rules

This section discusses heuristic rules for mashup query-tree optimization. The discussed heuristic rules are as follows.

- To carry out filter as soon as possible, this because of its execution can save the cost of following operators to several magnitudes. Generally, filter causes the intermediate results for computation much smaller. But in mashups business logic, if truncate and tail are prior to filter, then not to change the order of filter operator.
- To execute filter and projection simultaneously. If there are several filter operators and projection operators, and they are all applied to the same data collection, then complete all the operators at once which can avoid duplication of scanning the data collection.
- To combine projection with union and join, this can reduce scanning times on data collections.

- To combine sort with join and unique, it can reduce scanning times on data collections. This is for doing sort operations on unique attributes and join attributes.

Based on the above-mentioned heuristic rules, the optimization rules for mashup query-tree to abide by are as follows. Optimization rule $<x, y>$ represents x before y:

(1) <filter, y>, y$\in${sort, unique, projection, join, union}
(2) <projection, union>, <projection, join>
(3) <union, unique>, <union, sort>
(4) <sort, join>, <sort, unique>

These rules can be proven by operator equivalent transformation rules and operator performance model. Following, we prove rule of <filter, sort> to illustrate the rationality and validity of the above-discussed rules.

Rule: Given data source D and filter and sort operator, where D consists of n tuples. It can be concluded that performing filter first can enhance the efficiency, that is, $t(s(\sigma(D)))<t(\sigma(s(D)))$.

Proof: According operator equivalent rule 2, we can acquire $\sigma(s(D))=s(\sigma(D))$, so switching filter and sort can obtain the same results; According to operator performance model, $t(s(\sigma(D)))=(c(\sigma)+s(\sigma)\times c(s))\times n$, it is because that in generally $0\leqq s(\sigma)<1$. Therefore, $t(s(\sigma(D)))<nc(\sigma)+nc(s)$; And $t(\sigma(s(D)))=(c(s)+s(s)\times c(\sigma))\times n$, as $s(s)=1$, then $t(\sigma(s(D)))=nc(s)+nc(\sigma)$. That is, $t(s(\sigma(D)))<nc(s)+nc(\sigma)=t(\sigma(s(D)))$. The original proposition has been proved.

Other rules can be proven by referring to this proof.

Following, we present mashup optimization algorithm, which follows above heuristic rules and operator equivalent transformation rules.

Algorithm: mashup performance optimization algorithm
Input: a mashup query-tree T
Output: the optimized mashup query-tree.
Steps:

1. Traverse T and check whether it exists operators sequence like $\sigma_F(D_1\cup D_2)$ and $\sigma_F(D_1\triangleright_j D_2)$, if exist, then use operator equivalent transformation rules 6 and 7 to convert them into the form of $\sigma_F(D_1)\cup\sigma_F(D_2)$ and $\sigma_F(D_1)\triangleright_j\sigma_F(D_2)$.
2. Traverse T, find filter operators in T. Transfer each filter operators to the tree's leaf apex as far as possible using the heuristic rule 1 and operator equivalent transformation rules 1, 2 and 3.
3. Traverse T, find projection operators in T. Apply operator equivalent transformation rules 5 and 8 and the heuristic rule 2 on each projection operator to move it to the tree's leaf apex as far as possible.
4. Traverse T, search adjacent filter operators sequence, projection operators sequence and pairs of filter and projection. Apply heuristic rules 1 and 2 and operator equivalent transformation rules 3, 9 and 10 on these sequences, convert these sequences to a single filter operator, projection operator or filter following projection operator, which enable several filter or projection operators execute simultaneously to reduce scanning times on the data set.

5. Traverse T, use heuristic rules 3 and 4 to optimize union, unique, sort and join operator sequences.

We perform optimization on Aggregated News Alerts with the above algorithm, and the optimization result is as shown in Fig. 2. The average execution cost of 100 request times for optimized Aggregated News Alert is 1436.3ms. Comparing with the non-optimized, the performance improvement is 13.40%. We expect there would be better when the fetched data is large-scale. Following, we construct an experiment to verify our algorithm can improve the run-time performance of mashups.
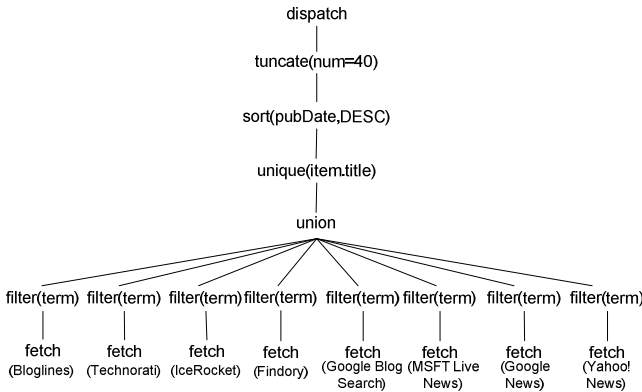

Fig. 2. The optimized query-tree of Aggregated News Alerts

5   Experiment and Discussion

The goal of this experiment is to estimate the effect of the optimization algorithm on mashups efficiency. First, we introduce the experiment design; next, we carry on analysis according to experiment results and give experiment conclusion.

Table 2. Test cases

No.	Case name	Logic structure	Description
1	YouTube Tags to Rss	fetch+filter+dispatch	Advanced YouTube filter pipe
2	Stock Quote Watch	fetch{2}+filter+join+dispatch	Yahoo finance stock quote watch list feed
3	Price Watcher	fetch{3}+truncate{3}+union+sort+dispatch	To buy and get the best price
4	World Weather Viewer	fetch{7}+union+projection+filter+dispatch	Provide world weather service
5	Twitter Monitor	fetch{4}+filter{4}+union+unique+sort+dispatch	A twitter observer
6	Aggregated News Alerts	fetch{8}+union+filter+sort+unique+truncate+dispatch	An aggregated news search service
7	Blog Search	fetch{6}+filter{4}+union+sort+unique+truncate+dispatch	Search blogs content feed
8	Meta Search Alerts	fetch{11}+filter{10}+union+sort+unique+dispatch	Union search service
9	Search Jobs	fetch{11}+filter{10}+rename{10}+union+dispatch	Jobs seek enquiry service
10	Hot Deals Search	fetch{10}+filter{11}+sort{10}+rename{2}+truncate{9} +unique{2}+union{2}+dispatch	Find hot deals service

We select 10 popular application cases from Yahoo! Pipes Community. The selected test cases are sorted in accordance with the logic complexity of the case (the number of operators), the test cases are shown in Table 2. We take Mashroom [6] as test platform, and take the case's execution time as test criteria.

First, we construct these 10 cases by Mashroom, and then compute average execution time of 100 request times for each case; subsequently, we perform performance optimization algorithm on each case, and compute average execution time of 100 request times for each case, the test results are shown in Fig. 3.

Fig. 3(a) represents the non-optimized and optimized average time for each case, we can conclude that when the mashups are relatively simple, the optimized results are not obvious, the reason is that when the mashups are simple, the end-users are easier to follow cost minimum rule to build mashup, but when the mashups become complicated, the end-users are hard to follow cost minimum rule, therefore the efficiency of optimized mashups has distinct enhancement. Fig. 3(b) represents the improved percentage of the efficiency of each case after applying optimization algorithm, the average improved percentage of the 10 cases is 12.87%. As illustrated in Fig. 3(b), with the increase of mashups complexity, the improved percentage also increases. When the mashup likes case 8, 9 or 10, the improved percentage comes to about 20% and tends to stable. Through analyzing the experimental results, we can draw a conclusion our presented strategy can reduce mashups execution times and enhance mashups run-time performance. We expect in high concurrent and large scale data environment, the strategy's optimization effect such as execution time and throughput would be better.
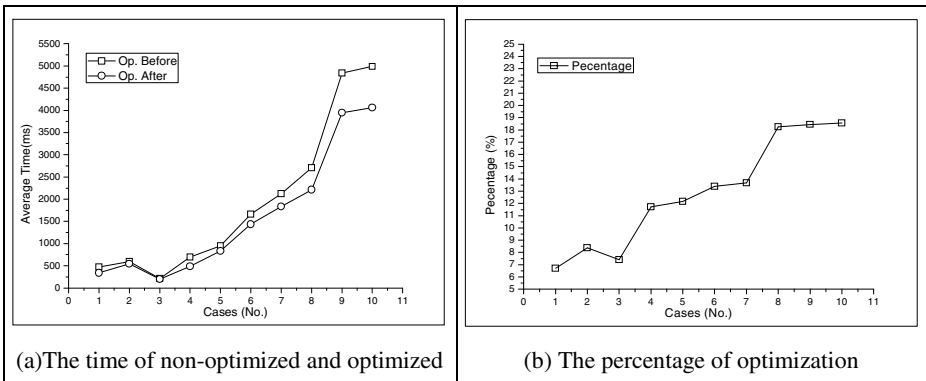

(a)The time of non-optimized and optimized

(b) The percentage of optimization

Fig. 3. The strategy evaluation

6   Conclusion

Although mashups technologies provide opportunities for end-users to make use of Internet-based resources to build personalized Web applications, mashups run-time performance is a new noticeable challenge. Therefore, we define mashup query-tree model and mashups operator performance model, and present operators equivalent transformation rules and mashup query-tree heuristic rules. On this basis, this paper proposes an optimization algorithm for mashups performance. Not only that, this paper conducts an experiment to test the impact of the algorithm on mashups efficiency, the

results show that this algorithm can improve the efficiency of mashups. In future works, we are ready to test the optimization effect in high concurrent and large-scale data environment.

References

1. Yu, J., Benatallah, B., Casati, F., Daniel, F.: Understanding Mashup Development. IEEE Internet Computing 12(5), 44–52 (2008)
2. Yahoo! Pipes: Rewire the web (2011), `http://pipes.yahoo.com/`
3. Altinel, M., Brown, P., Cline, S., et al.: Damia-A Data Mashup Fabric For Intranet Applications. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 1370–1373 (2007)
4. Wong, J., Hong, J.I.: Making Mashups with Marmite:Towards End-User Programming for the Web. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1435–1444 (2007)
5. Kongdenfha, W., Benatallah, B., Saint-Paul, R., Casati, F.: SpreadMash: A Spreadsheet-Based Interactive Browsing and Analysis Tool for Data Services. In: Bellahsène, Z., Léonard, M. (eds.) CAiSE 2008. LNCS, vol. 5074, pp. 343–358. Springer, Heidelberg (2008)
6. Wang, G., Yang, S., Han, Y.: Mashroom: End-User Mashup Programming Using Nested Tables. In: Proceedings of the 18th International Conference on World Wide Web, pp. 861–870 (2009)
7. Silberschatz, A., Korth, H.F., Sudarshan, S.: Database System Concepts, 5th edn. (Simplified Chinese Translation Edition), pp. 378–400. The McGraw-Hill Education (Asia) Co. China Machine Press (2006)
8. EditGrid (2011), `http://www.editgrid.com/`
9. Ennals, R.J., Garofalakis, M.N.: MashMaker: Mashups for the Masses. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, pp. 1116–1118 (2007)
10. Hartmann, B., Wu, L., Collins, K., et al.: Programming by a Sample: Rapidly Creating Web Applications with d.mix. In: Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, pp. 241–250 (2007)
11. Di, L.G., Hacid, H., Paik, H., et al.: Data Integration in Mashups. ACM SIGMOD Record 38(1), 59–66 (2009)
12. Hassan, O.A., Ramaswarny, L., Miller, J.A.: MACE: A Dynamic Caching Framework for Mashups. In: Proceedings of the IEEE International Conference on Web Services, pp. 75–82 (2009)
13. Hassan, O.A., Ramaswarny, L., Miller, J.A.: Enhancing Scalability and Performance of Mashups Through Merging and Operator Reordering. In: Proceedings of the IEEE International Conference on Web Services, pp. 171–178 (2010)
14. Schek, H.J., Scholl, M.H.: The relational model with relation-valued attributes. Information Systems 11(2), 137–147 (1986)
15. Babcock, B., Babu, S., Datar, M., et al.: Operator scheduling in data stream systems. The International Journal on Very Large Data Bases 13(4), 333–353 (2004)
16. Abadi, D.J., Carney, D., Cetintemel, U., et al.: Aurora: a new model and architecture for data stream management. The VLDB Journal 12(2), 120–139 (2003)

Model-Based Similarity Measure in TimeCloud

Thanh-Nguyen Ngo[1], Hoyoung Jeung[2], and Karl Aberer[1]

[1] École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{thanhnguyen.ngo,karl.aberer}@epfl.ch
[2] SAP Research, Brisbane, Australia
hoyoung.jeung@sap.com

Abstract. This paper presents a new approach to measuring similarity over massive time-series data. Our approach is built on two principles: one is to parallelize the large amount computation using a scalable cloud serving system, called TimeCloud. The another is to benefit from the filter-and-refinement approach for query processing, such that similarity computation is efficiently performed over approximated data at the filter step, and then the following refinement step measures precise similarities for only a small number of candidates resulted from the filtering. To this end, we establish a set of firm theoretical backgrounds, as well as techniques for processing kNN queries. Our experimental results suggest that the approach proposed is efficient and scalable.

Keywords: similar measure, time-series, cloud computing.

1 Introduction

As time-series data becomes ubiquitous, the demand for storing and processing massive time-series in the cloud is growing rapidly. To meet this demand, the LSIR laboratory[1] has been developing a storage-and-computing platform for managing large volumes of time-series in the cloud. TimeCloud is established upon a combination of several cloud systems, such as Hadoop [2] and HBase [3], while gearing various novel approaches towards significantly boosting the performance of large-scale data analysis on distributed time-series. One of the novel features in TimeCloud is to employ *model-based views*—which are database views approximating time-series using well-established models—for efficient data processing. In this paper, we present one mechanism that lies in the core of the feature, i.e., similarity measure of distributed time-series managed in the model-based views.

Measuring a similarity is a fundamental operation in a wide range of applications that process temporally ordered data, such as stock prices, sensor readings, trajectories from moving objects, and scientific data. Despite the importance of similarity measure, computing similar time-series over a large volume of data still remains as a difficult problem. Although a rich body of previous studies

---

[1] http://lsir.epfl.ch

have dealt with efficient computation of time-series [4, 1, 5], their proposals become limited when the data volume grows.

In this paper, we present a very different approach from the existing works. The key differences are twofold: First, we parallelize the computation using multiple nodes (servers), taking advantages of the cloud serving systems Hadooop and HBase. These systems make TimeCloud scale-out, allowing us to deal with huge volumes of time-series data. Second, we apply the well-known *filter-and-refinement approach* [6] for measuring similarities across different nodes. Specifically, we first approximate a given time-series using an either constant or linear model, and then store the approximated data into a model-based view. Given multiple model-based views containing approximated time-series data, we then find a candidate set that potentially satisfies a given query condition, while facilitating very efficient processing over the model-based views. For each candidate, we next measure an accurate similarity using full-precision data to validate the result.

In order to embody the approach, we establish theoretical foundations that can serve as the basis in computing distances between approximated time-series stored in the model-based views. As we deal with two different approximation models, the computation for similarity measure requires all pairs of different model-approximated data. Obviously, this needs very firm, non-trivial foundations, which we present in this paper. Furthermore, we offer details for processing kNN queries while taking the advantage of the filter-and-refinement approach. The beauty of our approach is to guarantee no false miss at query results, as the query processing technique is built on the foundations. In our experimental study, we will analyze the effect of this approach while applying different parameter settings.

The remainder of the paper is organized as follow: Section 2 offers a set of definitions, and establishes the theoretical foundations for the kNN query process presented in Section 3. We then discuss about experimental results in Section 4, and conclude in Section 5.

2   Similarity Measure over Model-Based Views

2.1   Definition

Definition 1 (Time-Series). *A time-series t of length n is a temporally ordered sequence $t = [t_1, \ldots, t_n]$ where point in time i is mapped to a d-dimensional attribute vector $t_i = (t_{i_1}, \ldots, t_{i_d})$ of values t_{i_j} with $j \in \{1, \ldots, d\}$. A time-series is called univariate for $d = 1$ and multivariate for $d > 1$.*

The work relies heavily on transformed models, and the existing system only converts univariate time-series. Therefore, we only consider the univariate time-series in the scope of the paper. If a time-series has multiple attributes, we consider it as multiple univariate time-series. From now on, when mentioning on time-series, we consider it as univariate time-series unless stated otherwise. In addition, we only consider time-series with the same interval.

Definition 2 (Common Points). *Two points of two time-series are called common if they occur at the same time.*

Definition 3 (Common Interval). *The common interval of two segments or two time-series is the greatest interval $[a, b]$ such that time a and b belong to both segments or time series. Two segments limited by the common interval are called common segments.*

With the Definition 3, two time-series may not have common segments if one time-series starts after another time-series ends. Or, the common segments of two time-series are time-series themselves if their starting points and end points are common.

Definition 4 (Euclidean Distance). *The Euclidean distance between two time-series is also the Euclidean distance of their common segments $s = [s_1, \ldots, s_n]$ and $t = [t_1, \ldots, t_n]$ of length n, and it is defined as:*

$$Eucl(s, t) = \sqrt{\sum_{i=1}^{n} (s_i - t_i)^2}$$

We can consider a time-series of length n as an n-dimensional point in space. The value at time t_i is mapped into the ith dimension. So, two common points will be mapped into the same dimension. And when evaluating the Euclidean distance between two points in the space, we only consider dimensions having two points of both time-series.

2.2 Model-Based Views

Since the major component of the back end consists of an HBase instance, the way the data is stored inside HBase becomes of major concern, not only for full precision data but also for the parameters used for model-based approximations. Fig. 1 represents in a schematic way how the data is organized in TimeCloud.

SensorID: Timestamp	Full Precision		Linear model		Constant model	
	temp	wind	temp'	wind'	temp"	wind"
x1	v1					
x2		v4				v13
x3						
x4	v2	v5		(v9, s2)	v11	
x5						
x6		v6				
x7	v3		(v8, s1)		v12	

Fig. 1. A snapshot of a model-based view

2.3   Calculating the Euclidean Distance over Models

Full Precision Model vs. Other Models. To determine the Euclidean distance between a full precision model and an another model, at first, we determine the common interval between these two time-series. Then, we apply the formula in Definition 4 to calculate the distance between every common points of two time-series. Finally, we square root the sum of all square of individual distances to get the Euclidean distance between these two time-series.

Constant Model vs. Constant Model. At first, we divide constant segments of two time-series into common segments. Then we calculate the distance of those common segments and then aggregating them. Since the data does not change within a common segment, the distance of common segments is equal to the distance of two common points multiply by the square root of number of common points in those segments.

When determining the common segments, we know the starting and end time of those segments, and we also know the interval of those time-series. Therefore, we can determine the number of common points of those common segments. In addition, we also know the values of these segments. Hence, we can determine the distance of these segments without aggregating all individual distances of common points.

Linear Model vs. Linear or Constant Model. At first, we evaluate the Euclidean distance of two time-series in linear models. As the implementation on constant models, we devise a similar algorithm to quickly return the Euclidean distance between two common linear segments. Assume the formula representing those segments are $y = ax + b$ and $y = cx + d$. Apply the formula in Definition 4, the square of the Euclidean distance of two common segments s, t with k common points is:

$$Eucl^2(s,t) = \sum_{i=1}^{k}(s_i - t_i)^2 = \sum_{i=1}^{k}((ax_i + b) - (cx_i + d))^2$$

$$= (a-c)\sum_{i=1}^{k} x_i^2 + 2(a-c)(b-d)\sum_{i=1}^{k} x_i + k(b-d)^2 \qquad (1)$$

Let t be the interval of two time-series, so $x_{i+1} = x_i + t$. We have:

$$k = \frac{x_n - x_1}{t} + 1 \qquad (2)$$

$$\sum_{i=1}^{k} x_i = \frac{k}{2}(x_1 + x_n) = \frac{x_1 + x_n}{2}\left(\frac{x_n - x_1}{t} + 1\right) \qquad (3)$$

$$\sum_{i=1}^{k} x_i^2 = \frac{1}{6t}[x_n(x_n + t)(2x_n + t) - x_1(x_1 - t)(2x_1 - t)] \qquad (4)$$

Replace (4), (3) and (2) into (1), we have:

$$Eucl^2(s,t) = \frac{a-c}{6t}[x_n(x_n+t)(2x_n+t) - x_1(x_1-t)(2x_1-t)]$$
$$+ (a-c)(b-d)(x_1+x_n)(\frac{x_n-x_1}{t}+1)$$
$$+ (b-d)^2(\frac{x_n-x_1}{t}+1) \tag{5}$$

Thus, similar to the implementation on constant models, we divide two time series in linear models into common segments. Then we calculate the square of the distance of common segments. The formula to determine this value only depends on the starting and end time, the coefficients of segments and the interval of those time-series.

A time-series in constant model is a special case of the linear model when the slope is equal to zero. So, we can apply the implementation on two linear models to calculate the distance of a time-series in linear model and a time-series in constant model.

2.4   Maximum Error of the Euclidean Distance of Two Time-Series

Definition 5 (Maximum Error Bound of Time-Series). *Given a time-series* $t = [t_1, \ldots, t_n]$ *and its representation* $t' = [t'_1, \ldots, t'_n]$ *in its model. The maximum error bound of t over its model is a value* $meb(t)$ *such that:*

$$|t_i - t'_i| \leq meb(t), \qquad \forall i = 0..n$$

In general, the value of maximum error bound of a time-series over its model is predefined. Then we construct the model such that it satisfies the formula in Definition 5 and we try to maximize the number of time-series points in a segment. With this approach, the number of segments in the model is minimized, and it is efficient to access and compute.

Definition 6 (Maximum Error Bound of Euclidean Distance). *Given time-series s and t and their representations* s', t' *in their models. The maximum error bound of the Euclidean distance between s and t over their models is a value* $MEB(s,t)$ *such that:*

$$|Eucl(s,t) - Eucl(s',t')| \leq MEB(s,t), \qquad \forall s', t'$$

From the Definition 6, the estimated distance between two time series differs from the real distance by an upper bound. Hence, when calculating the Euclidean distance on models, we do not know exactly the distance between two time-series, but we can estimate the range in which the distance belongs to.

Before giving a formula to determine the value of the maximum error bound of the Euclidean distance between two time-series, we need to prove the following lemma.

Lemma 1. *Given two time-series s, t and their representations s', t' in their models. Assume the common segments of s and t have n time series points. Then,*

$$||s_i - t_i| - |s'_i - t'_i|| \leq meb(s) + meb(t), \forall i = 1..n$$

Proof. Based on the Definition 5, $\forall i = 1..n$, we have:

$$- meb(s) \leq s_i - s'_i \leq meb(s) \tag{6}$$
$$- meb(t) \leq t_i - t'_i \leq meb(t) \tag{7}$$

Without loss of generality, assume $s_i \geq t_i$, let (6) - (7):

$$(s_i - t_i) - (s'_i - t'_i) \leq meb(s) + meb(t)$$
$$\Rightarrow |s_i - t_i| - |s'_i - t'_i| \leq meb(s) + meb(t) \tag{8}$$

Because of the equality in the role of t_i and t'_i in Definition 5, we also have:

$$|s'_i - t'_i| - |s_i - t_i| \leq meb(s) + meb(t) \tag{9}$$

From (8) and (9), we have:

$$||s_i - t_i| - |s'_i - t'_i|| \leq meb(s) + meb(t)$$

$\square$

Theorem 1 (MEB(s,t)). *Given two time-series s, t and their representations s', t' in their models. Assume the common segments of s and t have n time series points. Then,*

$$MEB(s,t) = \sqrt{n}(meb(s) + meb(t))$$

Proof. Let $d_i = s_i - t_i$, $d'_i = s'_i - t'_i$, and $m = meb(s) + meb(t)$
From the Lemma 1, we have:

$$\sum_{i=1}^{n} d'^2_i \leq \sum_{i=1}^{n} (d_i + m)^2$$
$$= \sum_{i=1}^{n} d^2_i + 2m \sum_{i=1}^{n} d_i + nm^2$$

Apply the Cauchy-Schwarz inequality $(\sum_{i=1}^{n} d_i)^2 \leq n \sum_{i=1}^{n} d^2_i$, we have:

$$\sum_{i=1}^{n} d'^2_i \leq \sum_{i=1}^{n} d^2_i + 2m \sqrt{n \sum_{i=1}^{n} d^2_i + nm^2}$$
$$= \left(\sqrt{\sum_{i=1}^{n} d^2_i} + \sqrt{n}m \right)^2$$
$$\Rightarrow Eucl(s', t') \leq Eucl(s, t) + \sqrt{n}(meb(s) + meb(t)) \tag{10}$$

Similarly, we also have:

$$\sum_{i=1}^{n} d_i^2 \leq \sum_{i=1}^{n} (d_i' + m)^2$$
$$\Rightarrow Eucl(s,t) \leq Eucl(s',t') + \sqrt{n}(meb(s) + meb(t)) \qquad (11)$$

From (10) and (11), we have:

$$|Eucl(s,t) - Eucl(s',t')| \leq \sqrt{n}(meb(s) + meb(t)) \qquad (12)$$

Apply the Definition 6, we have:

$$MEB(s,t) = \sqrt{n}(meb(s) + meb(t))$$

$\square$

The equality in (12) occurs when common segments of two time-series are parallel and $d_i' = d_i - m, \forall i = 1..n$ or $d_i' = d_i + m, \forall i = 1..n$. This condition may occur in theory, but it does not exist in our implementation, because we try to minized the total distance of raw data and its model.

3   KNN Processing

In this section, we introduce our implementation on solving the similarity measure problem using the filter-and-refinement method. At first, in the filter stage, we calculate the Euclidean distances of all time-series and the query time-series on models, we get the approximate distances and their maximum error bounds. From those values, we build a minimum candidate set which contains the result set. Then, we apply the refinement stage in which we calculate the true Euclidean distance between all time series in the candidate set and the query time-series to return exactly k nearest neighbors of the query time-series.

3.1   The Filter Stage

Given a query time-series q, and a database with n time-series $t_1, \ldots, t_n$, we aim to find k time-series from the database that are closest to q. To this end, we first compute the candidate set which definitely satisfy the given query condition.

Theorem 2. *Let t_i' and q' be representations of t_i and q in their models respectively. Let d_i' be the distance between t_i' and q' with the maximum error e_i. Let $a_i = d_i' - e_i$ and $b_i = d_i' + e_i$. Without loss of generality, assume $b_1 \leq \ldots \leq b_n$. The candidate set $S = \{t_i | a_i \leq b_k\}$ contains k nearest time-series of q and is minimal.*

Proof. First, we prove that S contains k nearest time-series of q. Take a time-series $t_j \notin S$, we need to prove that t_j is not one of k nearest neighbors of q. Since $t_j \notin S$, we have:

$$Eucl(t_j, q) \geq a_j > b_k$$

We also have:

$$Eucl(t_i, q) \leq b_i \leq b_k \qquad \text{with } i \leq k$$

Hence:

$$Eucl(t_j, q) > Eucl(t_i, q) \qquad \text{with } i \leq k$$

Because the true distance from t_j to q is greater than from k other time series, t_j will not belong to the result set.

Now, we prove that the set S is minimal. Let S' be the candidate set that contains k nearest time-series of q and is minimal. If $S \neq S'$, then $S \backslash S' \neq \emptyset$. Take $t_j \in S \backslash S'$.

Consider the following case: $Eucl(t_i, q) = \begin{cases} a_i, & \text{if } i = j \\ b_i, & \text{otherwise} \end{cases}$

Because $t_j \in S \Rightarrow a_j \leq b_k \leq b_i$ with $i \geq k$

$\Rightarrow Eucl(t_j, q) \leq Eucl(t_i, q)$ with $i \geq k$

$\Rightarrow t_j$ is one of k nearest neighbours of q

$\Rightarrow S'$ does not contain k nearest time-series of q,

contradicts to the assumption

Therefore, $S = S'$. $\qquad\qquad\square$

From Theorem 2, we calculate the Euclidean distance of all time-series and the query time-series in their models to retrieve the candidate set. This calculation is much faster than calculating the true distance of all time-series in the database because the time-series in their models have smaller number of segments.

3.2   The Refinement Stage

Given a query time-series q, and a candidate set S with m time-series $t_1, \ldots, t_m$. Our problem is to find k time-series from S that are closest to q.

Theorem 3. *Let t'_i and q' be representations of t_i and q in their models respectively. Let d'_i be the distance between t'_i and q' with the maximum error e_i. Let $a_i = d'_i - e_i$ and $b_i = d'_i + e_i$. Without loss of generality, assume $a_1 \leq \ldots \leq a_m$. The set $R = \{t_i | b_i \leq a_{m-k+1}\}$ is a subset of the result set.*

Proof. Take a time-series $t_j \in S$, we need to prove that t_j is one of k nearest time-series of q. We have:

$$Eucl(t_j, q) \leq b_j \leq a_i \leq Eucl(t_i, q) \qquad \text{with } i \geq m - k + 1$$

Therefore, we cannot find k time-series in S such that their distances to q are strictly smaller than the distance from t_j to q. In addition, the set S contains k nearest time-series of q, so t_j is one of k nearest time-series of q. $\qquad\square$

Based on the Theorem 3, at first, we retrieve time-series that definitely belong to the result set to not waste time to calculate the distances between them and the query time-series. Then we use the full precision model to calculate the true distances from the remaining time-series in the candidate set and the query time-series to determine the result set.

4   Experiments

All the experiments were executed on 2.4GHz Intel Core2 Quad CPU running Java implementation on Ubuntu 10.10 and the following parameters are used as default unless stated otherwise: length of time-series $l = 512$, number of nearest neighbors $k = 10$, error ratio $e = 3\%$, and number of time-series in the database $N = 1,000$.

4.1   Model-Based View Construction

At first, we evaluate the reduction of number of entries of time-series in model-based views on different error ratios. The result in Fig. 2 presents the total number of entries of all time-series in the database with respect to their models. It shows that the linear model always has smaller entries than the constant model, and of course, the full precision model is always the largest one. With respect to error ratios, the number of entries on the linear model are 27.55% with $e = 0.55\%$, up to 5.4% with $e = 5\%$ compared to the number of entries of the full precision model. The corresponding values on the constant model are 50.3% and 9.0% respectively.
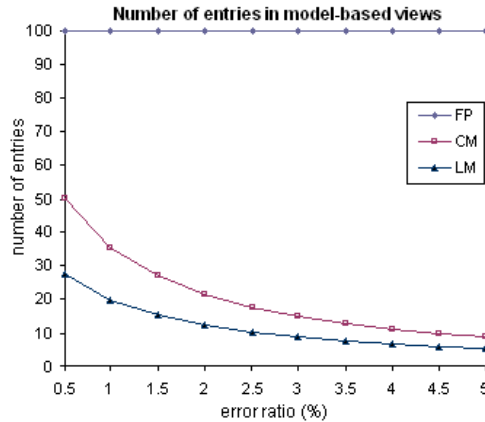

Fig. 2. Number of entries in model-base view on different error ratios

4.2   Effect of Maximum Error Ratios

In this experiment, we evaluate the effect of the maximum error ratio on the query processing time of the similarity measure problem. We evaluate it on three approaches: (1) running on the full precision model without using improvement technique, (2) using the filter-and-refinement method on the constant model, and (3) using the filter-and-refinement method on the linear model.

As depicted in Fig. 3, the linear model is always the fastest model in query processing and then is the constant model. When not using optimization technique,

the response is too long. In addition, the experiment shows that the performance peaks when the error ratio is 4% for the linear model and 4.55% for the constant model. The reason is that it takes much time on the filter stage if the error ratio is too small, and it takes much time on the refinement stage if the error ratio is too large. Therefore, choosing the appropriate error ratio is crucial to improve the system performance.
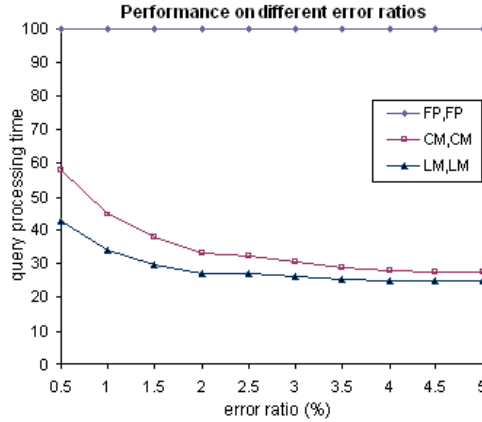

Fig. 3. Effect of the maximum error ratio on the query processing time

4.3   Effect of Number of Nearest Neighbors

In this experiment, we evaluate the effect of the number of nearest neighbors on the query processing time. Similar to the experiment in Sect. 4.2, we evaluate on three models and the result is depicted in Fig. 4.

The figure shows that it takes slightly more time to process the query if we increase the number of nearest neighbors. And this affects both constant and linear models. This is because when we increase the number of nearest neighbors, after the filter stage, the candidate set will be larger, and it takes more time in the refinement stage to calculate the real distance of time-series in the candidate set.

4.4   Effect of Number of Time-Series

In the last experiments, we evaluate the effect of the number of time-series in the database on the query processing time. The result depicted in Fig. 5 shows that the processing time of the converted models decreases when the number of time-series in the database increase. The reason is that the number of time-series in the candidate set does not increase linearly as the size of the number of time-series. Therefore, the refinement stage takes less time when we enlarge the database.
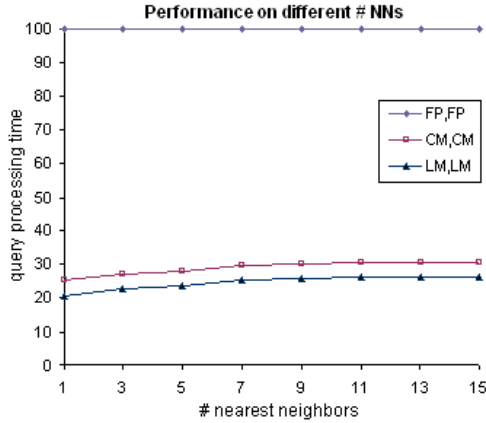
Fig. 4. Effect of the number of nearest neighbors on the query processing time
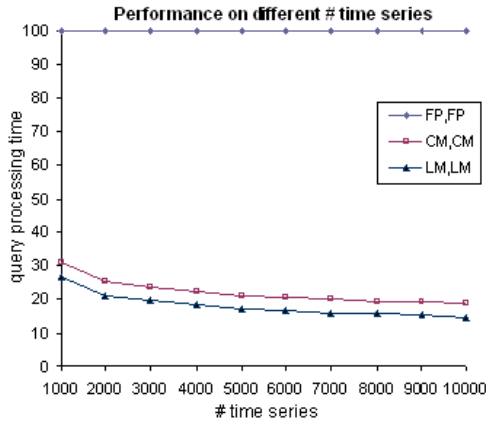

Fig. 5. Effect of the number of time-series on the query processing time

5   Conclusion

In the paper, we provide an efficient approach to processing kNN queries based on model-based similarity measures. To this end, we have established a set of important theoretical foundations for approximated time-series data processing. We then presented our query processing mechanisms built on the filter-and-refinement approach. The experiments showed that our approach runs more than three times faster than straightforward processing, while facilitating scalability of the computation using the TimeCloud system.

References

[1] Agrawal, R., Faloutsos, C., Swami, A.N.: Efficient Similarity Search in Sequence Databases. In: Lomet, D.B. (ed.) FODO 1993. LNCS, vol. 730, pp. 69–84. Springer, Heidelberg (1993)

[2] Apache. Hadoop. Website, `http://hadoop.apache.org/`

[3] Apache. Hbase. Website, `http://hbase.apache.org/`

[4] Chan, F.K.-P., Fu, A.W.-C., Yu, C.: Haar wavelets for efficient similarity search of time series: with and without time warping. IEEE Trans. on Knowl. and Data Eng. 15, 686–705 (2003)

[5] Korn, F., Jagadish, H.V., Faloutsos, C.: Efficiently supporting ad hoc queries in large datasets of time sequences. In: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, SIGMOD 1997, pp. 289–300. ACM, New York (1997)

[6] Seidl, T., Kriegel, H.-P.: Optimal multi-step k-nearest neighbor search. SIGMOD Rec. 27, 154–165 (1998)

Guess What I Want: Inferring the Semantics of Keyword Queries Using Evidence Theory

Jia-Jian Jiang, Zhi-Hong Deng⋆, Ning Gao, and Sheng-Long Lv

Key Laboratory of Machine Perception (Ministry of Education),
School of Electronic Engineering and Computer Science, Peking University
{jjj,ninggao}@pku.edu.cn, zhdeng@cis.pku.edu.cn
davidfracs@gmail.com

Abstract. The tagged and nested structure of an XML document provides quite detailed information about its structure and semantic, which is neglected by traditional keyword search model like TF-IDF and BM25 etc. Popular XML search models such as SLCA and XRANK tend to return the "deepest" node containing all given keywords, which usually leads to semantic loss. In this paper, we introduce the concept of *belief* in D-S evidential theory to evaluate primary search results, and propose a novel ranking model XSRET to rank them. In XSRET, We utilize XML's rich tag system to predict the semantics of keyword queries. For evaluating our SLCA-E model, we compare it with some state-of-the-art models, such as XSeek and XReal, and experimental result shows that XSRET outperforms these models. In addition, XSRET won the championship in the contest of data-centric track of INEX 2010.

1 Introduction

In traditional search model tf-idf [1][11], documents are defined to be search results, and ranking functions are based on relevance between documents and queries. Tf-idf model focuses on appearance frequency of a keyword while neglecting its semantics, even if it provides different information in different appearances. Therefore, if a keyword appears repeatedly in different documents or in different positions of the same document, their difference will be neglected, even if they actually have different meanings (e.g. *computer virus* and *Biological virus*). Tf-idf model uses only appearance frequency(term frequency and inverse document frequency) of a term to evaluate its weight, which is limited sometimes. For instance, fig.1 shows two result documents of query "Yimou", and document (a) is a segment of introduction to *Yimou Zhang* while document (b) is a segment of introduction to *Leung Ka Fai*. Judged by tf-idf model, document (b) will be a better answer than document (a), since "Yimou" appears three times in document (b) while only once in document (a). However, in fact, document (a) is a better result document to query "Yimou" than document (b). So why is that? Notice, all three appearances of "Yimou" in document (b) is in

---

⋆ Corresponding author.

```
<person>                                          <person>
 <name>Yimou Zhang</name>                          <name>Tony Leung Ka Fai</name>
 <overview>                                         <overview>
  <birth_date>1951</birth_date>                      <birth_date>1962</birth_date>
  <hometown>Xi'an, Shanxi</hometown>                 <hometown>Hong Kong</hometown>
 </overview>                                        </overview>
 <filmography>                                      <filmography>
  <direct>                                           <act>
   <movie>                                            <movie>
    <title>Beijing 2008 Olympics Games Opening Ceremony </title>  <title>20 30 40</title>
    <year>2008</year>                                  <year>2004</year>
   </movie>                                             <character>[Shi-Jie 'Jerry' Zhang]</character>
   <movie>                                            </movie>
    <title>Da hong deng long gao gao gua</title>     <movie>
    <year>1991</year>                                  <title>A1 tou tiao</title>
   </movie>                                             <year>2004</year>
   <movie>                                              <character>[Chief Editor Terrence Tsang Tat-si]</character>
    <title>Hong gao liang</title>                    </movie>
    <year>1987</year>                                 <movie>
   </movie>                                             <title>Ai zai bie xiang de ji jie</title>
   <movie>                                              <year>1990</year>
    <title>Huozhe</title>                              <character>[Zhao Nansan]</character>
    <year>1994</year>                                </movie>
   </movie>                                          </act>
   <movie>                                          </filmography>
    <title>Jin ling shi san chai</title>           <personal_quotes>
    <year>2011</year>                                <quote>
   </movie>                                            ... At that time, Yimou gave me a choice of playing the narrator,
   <movie>                                             like Leslie  Cheung did in Ashes of Time[1994]. In the end, I did
    <title>Ju Dou</title>                              what Yimou wanted and played Broken Sword,  who is the lover
    <year>1990</year>                                  of Flying Snow, played by Maggie. ."
   </movie>                                            (On working with Zhang Yimou in the movie, Hero)
  </direct>                                          </quote>
 </filmography>                                     </personal_quotes>
</person>                                          </person>
```

(a) Introduction to *Zhang Yimou*   (b) Introduction to *Leung Ka Fai*

Fig. 1. Two result documents when query="Yimou"

tag *quote*, that is, it is just mentioned by *Leung Ka Fai* for three times. Thus, tags in XML documents imply the semantic information of content under them, which should not be neglected.

Unstructured data such as HTML is a presentation language and hence cannot capture semantic information. XML data model addresses this limitation by allowing for extendable element tags, which can be arbitrarily nested to capture additional semantics. An XML document is organized by nested tags and the content between them, and tags imply semantics of the content under them. An XML document can be regarded as a tree, in which each node is a tag or content in the original XML document. In [2] [3] [4], nodes in an XML tree is classified into four types, they are entity node, connection node, attribute node and value node, and we continue to use it in this paper.

As discussed above, different appearances of a keyword in different documents or different positions of the same document have different semantics, while it is neglected by traditional research. In this paper, we focus on utilizing rich tags in XML datasets to infer semantics of keyword queries in XML search. Meanwhile, we exploit D-S evidential theory, which is a mathematical theory of evidence, to accomplish it. Our main contributions are summarized as follows:

1) This is the first work that utilizes evidence theory to XML keyword search, and we introduce the concept of *belief* to evaluate search results.

2) We utilize the tag system in XML datasets to infer the semantics of each keyword in given query, and explain it as the belief of given keyword to an attribute.

3) We promote a novel rank model XSRET to rank primary matches of given query, which is based on Dempster's rule of combination in D-S evidential theory.

The rest of the paper is organized as follows. We present preliminary on data model and D-S evidential theory in Section 2. Section 3 infers the semantics of given queries, and section 4 presents our rank model XSRET. Experimental evaluation is given in Section 5 and we conclude in Section 6.

2   Preliminaries

2.1   Data Structure

Xseek[1] classifies nodes in XML documents into four categories: entity, attribute, value, and connection. Among them, an entity can be regarded as a integrated unit that refers to something (a book, a person, a company, etc.), while attribute and value refer to the name and value of an attribute respectively , and connection refers to relationship between two entities (in XML documents, a connection node usually appears at the joint of some homonymous entities). For XML documents that obey a DTD profile, [1] makes the following inferences on node categories.

1) A node represents an entity if it corresponds to a *-node in the DTD.

2) A node denotes an attribute if it does not correspond to a *-node, and only has one child, which is a value.

3) A node is a connection node if it represents neither an entity nor an attribute. A connection node can have a child that is an entity, an attribute, or another connection node.

For instance, fig.2 shows the tree-style structure of an XML document, together with a DTD schema it obeys. In fig.2, *person* and *movie* are entities, which have both attributes and entities as their child nodes. *Overview*, *filmography* and *act* are connections, which connect several entities. *name*, *birth_date*, *biography*, *title*, *year*, and *character* are attributes, which have only a value node as their child. All the remaining leaf nodes are values. In this paper, when a keyword k appears in an attribute node A(represented by its name) or its son node(value node), then we say keyword k is in attribute A.

2.2   Search Result Definition

[12][13][14]firstly exploit the statistics of underlying XML database to address search intention identification, and [13] proposes object-level matching semantics called Interested Single Object (ISO) and Interested Related Object (IRO) to capture single object and multiple objects as user's search targets respectively. In this paper, we use the SLCA-based semantic model SLCA-E to define the result.
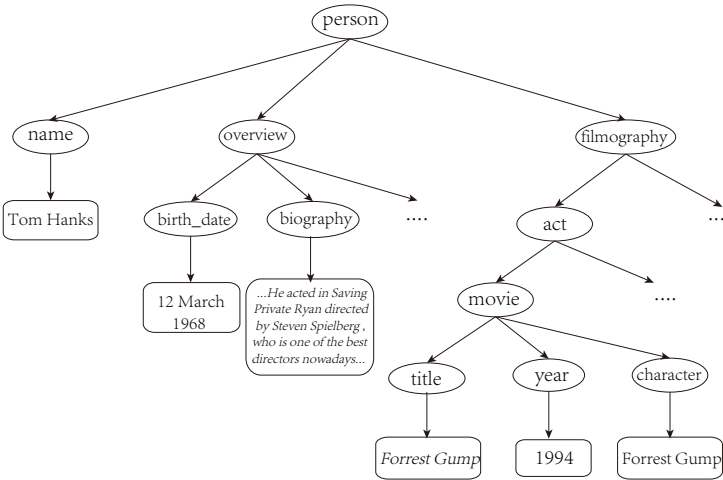
```
<!ELEMENT person (name, overview?, filmography?)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT overview (birth_date?, biography?)>
<!ELEMENT birth_date (#PCDATA)>
<!ELEMENT biography (#PCDATA)>
<!ELEMENT filmography (act?, direct?, write?)>
<!ELEMENT act (movie+)>
<!ELEMENT direct (movie+)>
<!ELEMENT write (movie+)>
<!ELEMENT movie (title, year, character?)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT character (#PCDATA)>
```

Fig. 2. A segment of tree-style XML document and its DTD Schema

VLCA models tend to return the "deepest" node containing the keywords, which usually leads to semantic loss. For instance, in fig.2, given a query "Tom Hanks", SLCA model returns the leaf node that contains "Tom Hanks" as result, which is confusing. In order to return a meaningful element, SLCA-E restricts the search result to be an entity, which is a fully-semantic unit of real world objects. An entity e is a tree or subtree rooted in an entity node, which is usually the root of a document tree.

Definition 1. *given a list of keywords $k_1, k_2, ,, k_n$ and an XML tree T, an answer to keywords $k_1, k_2, ,, k_n$ is an entity e that contains at least one instance of each keyword, meanwhile no other entity below it is an answer entity(of keywords $k_1, k_2, ,, k_n$).*

[4] Proposed two efficient algorithms to compute the SLCAs of a keyword query. When we compute SLCA-Es of a keyword query we just have to compute its SLCAs first, and then check each element in the set unless it is an entity. If any

element in SLCAs is not an entity node, we will visit its parent and replace the prior element with its parent if its parent is an entity. Otherwise we will visit the parent of its parent repeatedly until we find an entity node.

2.3   D-S Evidential Theory

The Dempster - Shafer evidential theory [10] is a mathematical theory of evidence. It allows one to combine evidence from different sources and arrive at a degree of belief (represented by a belief function) that takes into account the available evidences.

In D-S evidential theory, frame of discernment $\Theta = \{\theta_1, \theta_2, \cdots, \theta_n\}$ is the set representing all possible states of a system under consideration, and $A \in 2^\Theta$ is a possible solutions. Evidence theory assigns a belief mass $m : 2^\Theta \to [0, 1]$ to each solution, which is called *Basic Probability Assignment(BPA)*, and the belief mass satisfies $m(\emptyset) = 0$ and $\sum_{A \subseteq \Theta} m(A) = 1$.

Furthermore, D-S evidential theory defines belief and plausibility of solution A as follow.

$$Bel(A) = \sum_{B \subseteq A} m(B) \tag{1}$$

$$Pl\,(A) = 1 - Bel\,(\bar{A}) = \sum_{B \subseteq \Theta} m(B) - \sum_{B \subseteq \bar{A}} m(B) = \sum_{B \cap A \neq \emptyset} m(B) \tag{2}$$

In D-S evidential theory, $[Bel(A), Pl(A)]$ forms the probability interval of belief on solution A, and particularly, when $\forall A, B \subseteq \Theta, A \bigcap B = \emptyset$, then $Bel(A) = Pl(A)$.

Dempster's Rule of Combination settled the problem how to combine evidence from difference sources. Suppose there exist n evidences and their corresponding mass functions, then their combination is calculated as follow.

$$\begin{cases} m_\oplus\,(A) = m_1 \oplus \cdots \oplus m_\mathrm{n}\,(A) = K^{-1} \sum_{A_1 \cap \cdots \cap A_n = A} m_1(A_1) \cdots m_n(A_n) \\ K = \sum_{A_1 \cap A_2 \cap \cdots \cap A_n \neq \emptyset} m_1(A_1) m_2(A_2) \cdots m_n(A_n) \end{cases} \tag{3}$$

3   Inferring the Semantics of Keywords in a Given Query

Given a query containing n keywords as follow.

$$Q = \{k_1, k_2, \cdots, k_n\}$$

As each keyword can appear in different attributes, what we focus on is computing belief of all these attributes to that keyword. Suppose that Θ_{k_i} is the frame of discernment of keyword k_i, which contains all attributes in which keyword k_i appears.

$$\forall k_i, \exists \Theta_{k_i} = \{A_{i1}, A_{i2}, \cdots, A_{im}\}$$

here we define every solution to be a set of a single attribute, i.e. $\{A_{ij}\}$, and we use A_{ij} to briefly represent a solution. Thus Θ_{k_i} is also the set of all solutions of keyword k_i. As all solutions of keyword k_i is given, what we have to do next is to compute the belief of each solution(attribute) A_{ij} to keyword k_i. Notice, given any two different attributes A_{ip} and A_{iq} which satisfy $A_{ip} \bigcap A_{iq} = \emptyset$, we have following conclusion.

$$\mathrm{Bel}\,(A_{ij}) = \mathrm{Pl}\,(A_{ij}) \tag{4}$$

Thus, the probability interval of belief on A_{ij} is strict to a point, and we can use $Bel(A_{ij})$ to represent the belief on attribute on attribute A_{ij}.

3.1   Mass Function Based on Frequency and Length

To compute the belief of an attribute to given keyword, we should firstly determine some evidences to evaluate it, together with mass functions based on these evidences. Evidences can be various properties, and in this paper, we choose frequency of attribute containing given keyword and length of attributes to be two evidences.

As in tf-idf model, term frequency gives a measure of the importance of a term t within a particular document d. Similarly, the more keyword k appears in attribute A, the more likely that keyword k is used to describe attribute A. For instance, keywords like *Zhang Yimou* and *Leung Ka Fai* have a rather high distribution in attribute *name* of entity *person*, while *documentary* and *animation* mostly appear in attribute *genre* of entity *movie*. Therefore, we choose the frequency distribution of a keyword in its corresponding attributes to be one evidence, that is the frequency of attribute A containing given keyword. Since mass function m should satisfy $\sum\limits_{A \subseteq \Theta} m(A) = 1$, thus, we define the mass function based on distribution as follow.

$$m_{\mathrm{freq}}\,(A_{ij}|k_i) = \frac{\mathrm{freq}\,(A_{ij}, k_i)}{\sum_j \mathrm{freq}\,(A_{ij}, k_i)} = \frac{\mathrm{freq}\,(A_{ij}, k_i)}{\mathrm{freq}\,(k_i)} \tag{5}$$

where $\mathrm{freq}\,(A_{ij}, k_i)$ refers to the frequency of appearance of keyword k_i in attribute A_{ij}.

In addition to frequency distribution, we choose length of attributes to be another evidence, since frequency distribution sometimes is not enough for determination. As the instance shown in figure 1, keyword "Yimou" appears three times in document (b) while only once in document (a), however, document (a) is a better result than document (b). Although "Yimou" appears three times in document (b), yet it appears in the quote of *Leung Ka Fai* instead of describing the property of him. Therefore, we can conclude that although a document may contain the given keyword, yet the keyword may possibly not used to refer its property.

It is difficult to judge whether a keyword refers to the property of the document containing it, but we can use the evidence of length to infer the probability. As shown in figure 1 and figure 2, there are many attributes in an entity, and some

of them is long text, while some others only contain few keywords. It is obviously that the longer an attribute is, the more redundant information it has, or in other words, the shorter an attribute is, the more explicit its keyword is. We define the mass function based on length of attributes as follow.

$$m_{len}\left(A_{ij}|k_i\right) = \frac{\left(len\left(A_{ij}\right)\right)^{-1}}{\sum_j \left(len\left(A_{ij}\right)\right)^{-1}} \tag{6}$$

where $len\left(A_{ij}\right)$ is average length of all attribute nodes A_{ij} that contains keyword k_i.

3.2   Computing the Belief of Attributes

Given mass functions based on frequency and length, we can combine them to compute the belief of attributes based on Dempster's rule of combination as follow.

$$\begin{cases} \mathrm{Bel}\left(A_{ij}|k_i\right) = m_{freq} \oplus m_{len}\left(A_{ij}|k_i\right) = \frac{1}{\Delta_i} m_{freq}\left(A_{ij}|k_i\right) \times m_{len}\left(A_{ij}|k_i\right) \\ \Delta_i = \sum\limits_{j=1}^{m} m_{freq}\left(A_{ij}|k_i\right) \times m_{len}\left(A_{ij}|k_i\right) \end{cases} \tag{7}$$

For instance, given a keyword "French" and four attributes which contain it. In pre-processing, we can obtain the information about total appearance frequency of "French" as well as attributes containing "French" and their average length. We can judge the belief of the four attributes to "French" as follow.

Table 1. BPA of attributes when keyword="French"

Attributes(A)	$m_{dist}(A|\text{"French"})$	$m_{len}(A|\text{"French"})$	$Bel(A|\text{"French"})$
language	38986/90196	6/7.8482	0.8809
name	5029/90196	6/13.8515	0.0645
character	12090/90196	6/36.1828	0.0544
biography	2071/90196	6/2300.91	0.0002

4   Computing the Belief of Entities

In our rank model XSRET(XML Search Ranking based on Evidence Theory), we computes the belief of each entity to the given query, and rank the entities based on their beliefs. We define Θ_Q as the frame of discernment of query Q, in which each entity e_i contains all keywords in query Q.

$$\forall Q, \exists \Theta_Q = E = \{e_1, e_2, \cdots, e_z\}$$

As queries are made up of several keywords, we compute the belief of each entity based on the belief of appearances of keywords in it. in this paper, we treat each keyword as an evidence, and define mass functions based on appearances of keywords in the entity. We define the belief of entity e_t to query Q as follow.

$$\begin{cases} \mathrm{Bel}\left(e_t|Q\right) = m_{k_1} \oplus m_{k_2} \oplus \cdots \oplus m_{k_n}\left(e_t|Q\right) = \frac{1}{\Lambda_t} \prod\limits_{i=1}^{n} m_{k_i}\left(e_t|Q\right) \\ \Lambda_t = \sum\limits_{t=1}^{z}\left(\prod\limits_{i=1}^{n} m_{k_i}\left(e_t|Q\right)\right) \end{cases} \tag{8}$$

where $m_{k_i}(e_t|Q)$ is the belief of entity e_t to query Q when considering keyword k_i, and we define it as follow.

$$m_{k_i}(e_t|Q) = \sum_{A_{ij} \in e_t \wedge k_i \in A_{ij}} Bel(A_{ij}|k_i) \qquad (9)$$

For instance, when query Q={USA, president, documentary}, to compute the belief of all entities that contain all three keywords, we firstly compute the sum of belief of all attributes to each keyword, and then add them together.

Table 2. BPA of documents when query={USA, president, documentary}

Entities(e_i)	$m_{"USA"}(e_i|Q)$	$m_{"president"}(e_i|Q)$	$m_{"documentry"}(e_i|Q)$	$Bel(e_i|Q)$
e_1	$Bel(country|"USA")$ $=0.4853$	$2Bel(title|"president")+$ $Bel(keyword|"president")$ $=1.6948$	$Bel(genre|"documentary")$ $=0.6915$	0.99635
e_2	$Bel(country|"USA")$ $=0.4853$	$Bel(trivia|"president")$ $=0.0062$	$Bel(genre|"documentary")$ $=0.6915$	0.00364
e_3	$Bel(plot|"USA")$ $=0.0004$	$Bel(plot|"president")$ $=0.0147$	$Bel(genre|"documentary")$ $=0.6915$	0.00001

In table 2, e_1 is a movie named *Portraits of Presidents: Presidents of a World Power*, which is a documentary movie mentioned about tens of American presidents, while e_2 is a movie named *Surviving the Eruption at Mt. Pinatubo* and e_3 is a movie named *Transpersonal Conversations: Frances Vaughan, Ph.D* , both of which are obviously irrelevant. The BPA of these three entities evaluate them as expected. The overall processing of our model XSRET is shown in fig.3.
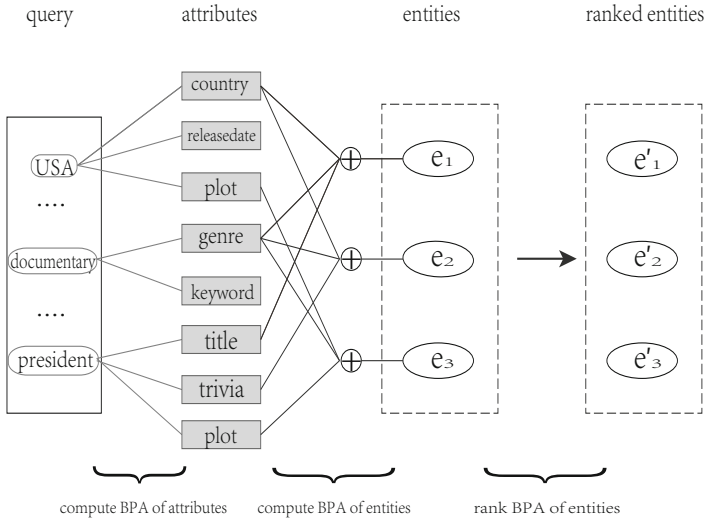
Fig. 3. Procedure example of XSRET model
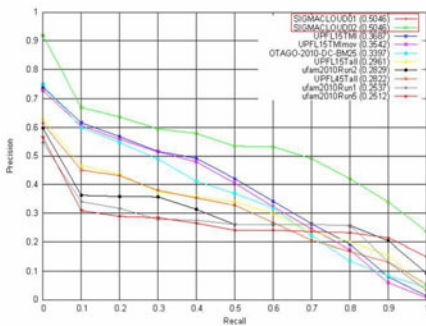
5   Experiment

5.1   Datasets

In this paper, we use the IMDB data collection from INEX 2011 Data-Centric track, which is newly built from www.imdb.com. It consists of information about more than 1,590,000 movies and people involved in movies, e.g. actors/actresses, directors, producers and so on. Each object is richly structured. The total size of these files is 21.9GB, and figure 2 shows a segment of these files.
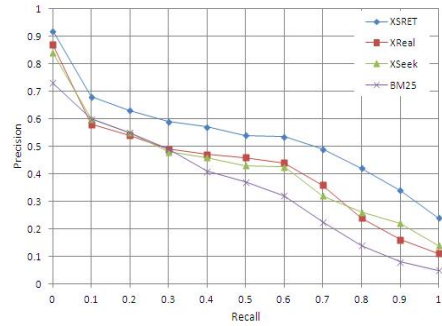
5.2   Results

We tested on all of the 28 queries from Data-Centric track topics presented in table 3(we neglected symbols like "", + and - here). Limited by space, we

Table 3. 28 topics in Data-Centric track of INEX 2010

Q_1	Yimou Zhang 2010 2009	Q_{15}	may the force be with you
Q_2	Dogme movies	Q_{16}	best director award Steven Spielberg
Q_3	stan lee actor	Q_{17}	Avatar James Francis Cameron
Q_4	brad pitt producer	Q_{18}	Stanley Kubrick movies director
Q_5	Shirley Temple	Q_{19}	Heath Ledger actor movies list
Q_6	Tom Hanks Ryan	Q_{20}	Movies directed Jean Pierre Jeunet Marc Caro
Q_7	Ingmar Bergman biography	Q_{21}	Movies Klaus Kinski actor movies good rating
Q_8	titanic jack rose	Q_{22}	Scarlett Johansson John Slattery
Q_9	hua mulan animation	Q_{23}	Comedy Woody Allen Scarlett Johansson
Q_{10}	director fearless jet li	Q_{24}	around the world in eighty days
Q_{11}	ancient Rome era	Q_{25}	tom hanks steven spielberg
Q_{12}	quentin tarantino thriller	Q_{26}	true story drugs addiction
Q_{13}	tom cruise movies	Q_{27}	making of The Lord of the Rings
Q_{14}	Clint Mansell composer	Q_{28}	romance movies by Richard Gere or George Clooney

(a) Best runs in Data-Centric track   (b) Comparison with other models

Fig. 4. Experiment results measured by MAP

briefly present the experimental results, and fig.4 shows the comparison result of our mode XSRET with other methods. In fig.4, (a) presents the performance of the whole document based MAP metric, and XSRET (SIGMACLOUD01 and SIGMACLOUD02) performs substantially better than the other runs at all recall points. In addition, we compare XSRET with latest models XReal and XSeek and present the result in fig.4(b), and XSRET gains a better result.

6   Conclusions

In this paper, we have a primary discussion about evaluating search results in XML keyword search based on D-S evidential theory. We utilize XML's rich tag system and structure information to infer the semantics of keyword queries, and we focus on analysis of each appearance of keyword in results. Experiments show that our rank model XSRET is more effective than the existing approaches.

References

1. Carmel, D., Maarek, Y.S., Mandelbrod, M., et al.: Searching XML documents via XML fragments. In: SIGIR 2003, pp. 151–158 (2003)
2. Liu, Z., Chen, Y.: Identifying meaningful return information for xml keyword search. In: SIGMOD Conference (2007)
3. Liu, Z., Walker, J., Chen, Y.: XSeek: A Semantic XML Search Engine Using Keywords. In: VLDB 2007, pp. 1330–1333 (2007)
4. Huang, Y., Liu, Z., Chen, Y.: eXtract: A Snippet Generation System for XML Search. In: VLDB 2008, pp. 1392–1395 (2008)
5. Xu, Y., Papakonstantinou, Y.: Efficient keyword search for smallest LCAs in XML databases. In: SIGMOD, pp. 537–538 (2005)
6. Guo, L., Shao, F., Botev, C., Shanmugasundaram, J.: XRANK: ranked keyword search over XML documents. In: SIGMOD (2003)
7. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a Core of Semantic Knowledge. In: WWW (2007)
8. Suchanek, F., Kasneci, G., Weikum, G.: YAGO: A Large Ontology from Wikipedia and WordNet. Journal of Web Semantics (2008)
9. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
10. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press (1976) ISBN 0-608-02508-9

11. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York (1986)
12. Bao, Z., Lu, J., Ling, T.W.: XReal: An Interactive XML Keyword Searching. Demo paper in CIKM (2010)
13. Bao, Z., Lu, J., Ling, T.W., Xu, L., Wu, H.: An Effective Object-Level XML Keyword Search. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5981, pp. 93–109. Springer, Heidelberg (2010)
14. Bao, Z., Ling, T.W., Chen, B., Lu, J.: Effective XML Keyword Search with Relevance Oriented Ranking. Full paper in ICDE (2009)

Re-ranking by Multi-modal Relevance Feedback for Content-Based Social Image Retrieval

Jiyi Li, Qiang Ma, Yasuhito Asano, and Masatoshi Yoshikawa

Department of Social Informatics, Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
`jyli@db.soc.i.kyoto-u.ac.jp`, {`qiang,asano,yoshikawa`}`@i.kyoto-u.ac.jp`

Abstract. With the recent rapid growth of social image hosting websites, it is becoming increasingly easy to construct a large database of tagged images. In this paper, we investigate whether and how social tags can be used for improving content-based image search results, which has not been well investigated in existing work. We propose a multi-modal relevance feedback scheme and a supervised re-ranking approach by using social tags. Our multi-modal scheme utilizes both image and social tag relevance feedback instances. The approach propagates visual and textual information and multi-modal relevance feedback information on an image-tag relationship graph with a mutual reinforcement process. We conduct experiments showing that our approach can successfully use social tags in the re-ranking of content-based social image search results and perform better than other approaches. Additional experiment shows that our multi-modal relevance feedback scheme significantly improves performance compared with the traditional single-modal scheme.

Keywords: CBIR, Image Re-ranking, Social Tags, Relevance Feedback.

1 Introduction

Social image hosting websites, e.g., Flickr, have recently become very popular. On these websites users can upload and tag their images and share them with others. This social tagging is similar to keyword annotation in traditional image retrieval systems. An important difference is that keyword annotation requires several experts for annotating images and requires too much time and labor if the image database is large. Social tagging does not have this problem, because a lot of users can participate in the tagging task. It is easier to construct a large database with tagged images. Another difference is that social tags are user-generated and folksonomy tags [1]. Compared with taxonomy keywords in keyword annotation, which use a number of specific fixed words, social tags have an open vocabulary. This results in social tags having lots of noises.

Social tags have been proven to be effective for providing and improving keyword-based image retrieval and are widely used on social image hosting websites. However, whether social tags are beneficial for improving content-based image retrieval (CBIR) has not been well investigated in existing work. CBIR

Fig. 1. Query by Example, Content-Based Similar Image Results and Social Tags

has a long history and a large amount of research has gone into it, but its performance still needs to be improved for practical application. In CBIR, for a query image sample, systems search for content-based similar images from a specific multimedia database by image visual information. Because the query image does not include any textual information, the relationships between the query image and the textual information of other images in the database are hard to evaluate because of the well-known semantic gap problem. For example, for the query "horse" image in Fig.1, it is difficult to determine the relationship between the query and the "cat" tag in the database. The effectiveness of social tags for improving content-based similar image results is, as yet, unknown.

In content-based similar image results of a given query image and database, relevant images are relatively few and irrelevant images are numerous. We observe that there is a characteristic followed by image semantics that relevant images are alike, whereas irrelevant images are diverse. For example, for a query "horse" image, its relevant images have the "horse" concept in common, whereas its irrelevant images have diverse concepts such as "cat", "bird", and so on. However in most cases content-based similar image results do not have this characteristic. Fig.1 shows a query image and its content-based similar images using SIFT feature[13]. These diverse images are regarded as "relevant" images in content-based similar image results by CBIR. On the other hand, social tags sometimes follow this characteristic. In Fig.1, the relevant images have common tag sets including a "horse" tag, while the irrelevant images have diverse tag sets. It shows that social tags may be able to be used for improving content-based image search results. However social tags do not always possess the above mentioned characteristic. A solution is to leverage user relevance feedback which can provide additional relevance information and reflect users' subjective intentions.

In traditional relevance feedback, a single media modal provides (and can only provide) limited information. Because both social image and tag information are available in our scenario, it is natural to propose a multi-modal relevance feedback scheme, which includes both social image and tag relevance feedback instances. In this kind of scheme users can label any modal and any number of

instances. The problem is how to design a flexible approach to leverage multi-model relevance feedback information in the re-ranking task.

We propose a supervised approach that re-ranks content-based social image search results. We construct an image-tag relationship graph model with both images and their tags as vertices, and in which image-tag annotation relationships are edges. Fig.1 gives a brief sample of one such graph. Our approach propagates visual and textual information on the graph with a mutual reinforcement process. In this process, the good tags (in bold) of relevant images contribute more scores on the graph; the bad tags of relevant images, and the good and bad tags (in italics) of irrelevant images contribute fewer scores on the graph; the relevant images contribute more to their tags, whereas the irrelevant images contribute less. We also develop a series of rules to propagate the multi-modal relevance feedback information on the graph and affect the rank scores of other social images and tags. After several iterations, the relevant images can obtain higher rank scores.

The main contributions of this paper are as follows.

- We investigate whether social tags can be used for improving content-based search results. To the best of our knowledge, this has not been well investigated in existing work. We successfully propose an approach that can use social tags to improve the results. It is independent of visual feature selections and is easy to be integrated with current CBIR systems. We conduct experiments showing that our approach performs better than other approaches.
- We propose a novel multi-modal relevance feedback scheme and develop an approach for this scheme. We conduct experiments showing that our multi-modal relevance feedback scheme can significantly improve performance compared with the traditional single-modal relevance feedback scheme. Our approach is effective and easy to implement. It provides a novel approach which propagates the relevance feedback information on the relationship graph.

The remainder of this paper is organized as follows. In Section 2 we give a brief review of related work. In Sections 3 we propose our approach. In Section 4 we report the experimental results. We give the conslusions in Section 5.

2   Related Work

There have been studies related to automatic image re-ranking for keyword-based image retrieval. One of the typical work is the well-known VisualRank approach proposed by Jing and Baluja [2] which applies a random walk method to image similarity complete graph for ranking images. On the other hand, there have been studies related to automatic image re-ranking for content-based image retrieval only based on visual features. For example, Hsu et al. [3] use clustering methods to adjust the rank results with the distance of a cluster from a query. However, we have different goals than the above mentioned studies. We concentrate on image re-ranking with social tags for content-based image retrieval. It has not been well investigated in previous work. The graph-based mutual reinforcement

approach and multi-modal relevance feedback we propose efficiently use both visual and textual information in the refining process.

On the other hand, relevance feedback (RF) has been widely used in image re-ranking in supervised scenarios. In early work, some approaches were proposed to adjust the weights of different components of the queries or change the query representation to better suit the user's information needs. Porkaew et al. [4,5] proposed query reweighting, query reformulation, query modification, query point movement and query expansion approaches. All of these approaches are feature-dependent and focus on the queries on feature space or the relationships among different features. Our proposed approaches are feature-independent and are directly based on the image similarity output by CBIR systems; this enables our approach to be easily integrated into current CBIR systems.

There have also been many studies focused on how to select RF instances for re-ranking. For example pseudo-relevance-feedback-based approaches [6,7] directly select instances from the search results as the RF instances. Hsu et al. [8] used clustering method to pseudo-RF instances for re-ranking text-based video search results. RF instance selection is not within the scope of our work. We show results to users and use the RF instances labeled by users. It is also useful if integrating these studies with our approach.

Furthermore many approaches use RF instances as training sets and utilize machine learning techniques to classify image search results. For example, Zhang et al. [9] and Chen et al. [10] proposed approaches using support vector machine (SVM). These approaches always include an offline learning process that uses many queries and their corresponding RF instances to learn a query-independent re-ranking model. However, they are not appropriate for our supervised re-ranking scenario. In our scenario, the approaches need to learn an online query-dependent model with real-time RF instances; there are few RF instances for training and it is possible that there are no negative instances. Our approach is query-dependent and still works well in our scenario.

3   Social Image Re-ranking

We introduce our social image re-ranking approach in this section. First, we define the terms and describe an image-tag relationship model, and then introduce the visual and textual descriptors. Thereafter we propose our multi-modal relevance feedback scheme and social image re-ranking approach in detail.

Our re-ranking task can be formulated as follows. For a given query image q, a content-based image retrieval system computes the top-n content-based similar image results $\mathcal{A} = \{a_1, \cdots, a_n\}$ from a social image database $\mathcal{D}$. Let s_{iq} be the similarity between q and a_i. We regard $\mathcal{A}$ as the candidate image set and the social tags of images in $\mathcal{A}$ as the candidate tag set $\mathcal{T} = \{t_1, \cdots, t_m\}$. We define $\mathcal{T}_{a_i}$ as the tag set of each image $a_i \in \mathcal{A}$. Users can select relevance feedback instances from $\mathcal{A}$ and $\mathcal{T}$. We define $\mathcal{RF}_\mathcal{A}$ and $\mathcal{RF}_\mathcal{T}$ as the social image and tag relevance feedback instance sets. Our task is to re-rank the image set $\mathcal{A}$ with both the tag set $\mathcal{T}$ and the relevance feedback set $\mathcal{RF}_\mathcal{A}$ and $\mathcal{RF}_\mathcal{T}$.

3.1   Image-Tag Relationship Model

To leverage both social image visual information and social tag textual information for re-ranking, we construct a graph model, shown in Fig.2 with the candidate sets $\mathcal{A}$ and $\mathcal{T}$ for analyzing image-tag relationships. The vertices of the graph model denote social images and their tags. Note that query image q contains no textual information.

The edges of the graph model denote the relationships among images and tags. There are three kinds of image-tag relationships: image-to-image relationship, tag-to-tag relationship, and image-to-tag annotation relationship. The first two kinds of relationships reflect the intra-relationships among images or tags. The third relationship reflects the inter relationship between images and tags.
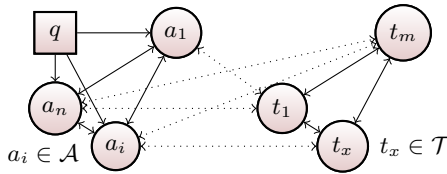
Fig. 2. Image-Tag Relationship Model

3.2   Visual and Textual Descriptor

To make use of visual and textual information in our approach, we convert them into visual and textual descriptors. The visual descriptors are based on image similarity. Our approach is baesd on these visual descriptors and independent of the selectin of visual features and CBIR algorithms, and thus easy to be integrated with current CBIR systems. In this paper, to compute image similarity, we use the following six types of low level features[13]: 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments and 500-D bag of words based on SIFT. The distance between image a_i and a_j on low level feature k is computed using Pearson correlation $d(\mathcal{H}_{ik}, \mathcal{H}_{jk})$ defined as

$$d(\mathcal{H}_{ik}, \mathcal{H}_{jk}) = \frac{\sum_x (\mathcal{H}'_{ik}(x) * \mathcal{H}'_{jk}(x))}{\sqrt{(\sum_y \mathcal{H}'_{ik}(y)^2) * (\sum_y \mathcal{H}'_{jk}(y)^2)}}, \mathcal{H}'_{ik}(x) = \mathcal{H}_{ik}(x) - \frac{\sum_y \mathcal{H}_{ik}(y)}{\mathcal{N}_k},$$

where $\mathcal{H}_{ik}$ and $\mathcal{H}_{jk}$ are feature vectors. $\mathcal{N}_k$ is the size of the feature vector k. The image similarity s_{ij} between two images with multi-feature is computed using a weighted sum.

$$s_{ij} = s(a_i, a_j) = \frac{\sum_k w_k d(\mathcal{H}_{ik}, \mathcal{H}_{jk})}{\sum_k w_k}.$$

We use $w_k = 1$ for any k in our work. This means that all low level features have the same weight. This strategy was often used in existing work such as [11]. For each image a_i in candidate image set $\mathcal{A}$, we propose a visual descriptor vd defined as $vd_i = s_{iq}$.

To leverage social tag information, we propose a tag importance measure tc_x for each tag t_x. This measure considers both the local tag frequency in candidate tag set $\mathcal{T}$ and the global tag frequency in database $\mathcal{D}$. It can evaluate how important tag t_x is to the current candidate tag set $\mathcal{T}$ in database $\mathcal{D}$. We propose a textual descriptor td defined as

$$tc_x = \frac{|t_x|_{\mathcal{T}}}{|t_x|_{\mathcal{D}}}, \quad td_x = \begin{cases} tc_x, \text{ if } |t_x|_{\mathcal{T}} > \delta, \\ 0, \text{ if } |t_x|_{\mathcal{T}} \leq \delta. \end{cases}$$

Here, $|t_x|_{\mathcal{T}}$ means the number of images in candidate tag set $\mathcal{T}$ that contain t_x, $|t_x|_{\mathcal{D}}$ means the number of images in database $\mathcal{D}$ that contains t_x. δ is a local frequency threshold for ignoring the noisy tags which have low frequency in $\mathcal{T}$ as well as in $\mathcal{D}$ and therefore have high value on tc_x.

3.3   Multi-modal Relevance Feedback

Relevance feedback has been proven to be an effective method for improving information retrieval performance. It can provide additional relevance judgments of the result and reflect a user's subjective intentions so as to generate more preferable results for the user. However, traditional single modal relevance feedback can only use limited multimedia information. Since both social image and tag information are available in our re-ranking scenario, we propose a multi-modal relevance feedback scheme, which takes into account both social image and tag relevance feedback instances.
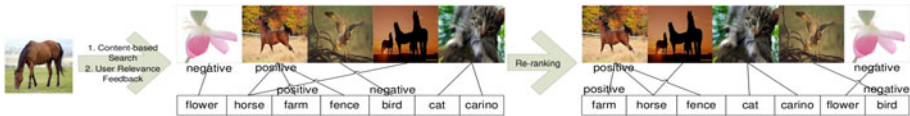

Fig. 3. Re-ranking with Multi-modal Relevance Feedback

Fig.3 shows a simple example of this scheme. In this scheme when the initial content-based image search results of the image queries are returned, users can label relevance feedback instances from the images in the results and the tags of these images as either positive or negative instances. Because users prefer finding the target images to checking the tag quality of an image, a tag instance labeled by a user is a tag that is relevant or irrelevant to the query image, and not to the image that this tag belongs to. Thereafter our re-ranking approach refines the results using this relevance feedback information. We execute the loop of labeling and re-ranking until users are satisfied. We define several terms in this re-ranking task: *Session* denotes a query task including the initial query and the relevance feedback rounds. *Round* is a relevance feedback process including labeling and re-ranking. *Iteration* denotes an iteration computation in our approach.

Traditional approaches to single-modal relevance feedback focus on single-modal information and do not provide solutions of using multi-modal information. We design a flexible approach, which has a basic mutual reinforcement

process for re-ranking and a series of additional rules for utilizing multi-modal relevance feedback information. The flexibility is mainly reflected in the ability of that users to label any modal, any number or any relevance type for instances, and our approach remains effective.

3.4   Basic Mutual Reinforcement Process

Following the image-tag annotation relationships in the graph model, we propagate the rank scores of images in $\mathcal{A}$ and tags in $\mathcal{T}$ along the links between images and tags. We observe a phenomenon that for an image a_i, when propagating the rank scores from images to tags, if a_i has a high rank score, its related tags will obtain higher rank scores. When propagating the rank scores is from tags to images, if the related tags of a_i have high rank scores, a_i will obtain a higher rank score. On the other hand, for a tag t_x, it also has similar phenomenon. Therefore, we naturally arrive at the following mutual reinforcement assumption: a high-ranked image for q is one to which many high-ranked tags point; a high-ranked tag for q is a tag that points to many high-ranked images. The iterative formulas for computing the rank scores are defined as follows:

$$Initialization: \mathcal{Q}'_0(a_i) = \Phi(vd_i), \ \mathcal{Q}'_0(t_x) = \Phi(td_x); \ 0 \leq \alpha, \beta \leq 1$$

$$Iteration: \begin{cases} \mathcal{Q}_{k+1}(t_x) = \alpha\Phi(td_x) + (1-\alpha)\sum_{\vee a_i:t_x\in T_{a_i}} \Phi(vd_i)\mathcal{Q}'_k(a_i) \\ \mathcal{Q}_{k+1}(a_i) = \beta\Phi(vd_i) + (1-\beta)\sum_{\vee t_x:t_x\in T_{a_i}} \Phi(td_x)\mathcal{Q}'_k(t_x) \\ \mathcal{Q}'_{k+1}(t_x) = \Phi(\mathcal{Q}_{k+1}(t_x)), \ \mathcal{Q}'_{k+1}(a_i) = \Phi(\mathcal{Q}_{k+1}(a_i)) \end{cases}$$

$$\Phi(\mathcal{Q}_k(a_i)) = \frac{Q_k(a_i) - min_j\{Q_k(a_j)\}}{max_j\{Q_k(a_j)\} - min_j\{Q_k(a_j)\}}, \ \Phi(\mathcal{Q}_k(t_x)) = \frac{Q_k(t_x) - min_y\{Q_k(t_y)\}}{max_y\{Q_k(t_y)\} - min_y\{Q_k(t_y)\}}$$

The iteration parameters α and β are damping factors. k is the number of iteration steps. We initiate $\mathcal{Q}'_0(t)$ of tags with textual descriptors and $\mathcal{Q}'_0(a)$ of images with visual descriptors. $\mathcal{Q}'_k(\cdot)$ is the normalized rank score of $\mathcal{Q}_k(\cdot)$ by using min-max normalization method.

For a candidate image a_i, its image similarity to the query image is an inherent property. Images that have high similarity can be regarded as more important on the graph. For a candidate tag t_x, it is also similar. Therefore we use visual descriptors and textual descriptors as the weights in the iterations. These weights represent the importance of these images and tags on the graph. As demonstrated in Section 4.2, these weights play an important role in performance enhancement.

3.5   Re-ranking with Relevance Feedback

We develop a series of rules to add to the basic mutual reinforcement process for utilizing multi-modal relevance feedback information. The graph model allows such modifications easy to implement. The key idea is that we mark relevance feedback instances on the graph, upgrade (downgrade) the score of positive (negative) instances, and propagate these scores through the links between images and tags, so that we can refine the rank scores of the related images and tags using the relevance feedback information. After several iterations, the relevance feedback information is propagated to all other related images and tags.

All these rules enforcedly change the value of some terms in the iteration formulas of the basic mutual reinforcement process, while the form of the iteration formulas are not changed. In the following rules, we use "+" to denote positive instances and "-" to denote negative instances.

Rule 1: At the beginning of each round, we use the rank results of the last round to initialize the rank scores. $\mathcal{Q}_0^{l+1} = \mathcal{Q}_h^l$, where l is the round number and h is the iteration number of the previous round.

Rule 2: At the beginning of each round, we change the value of the visual and textual descriptors of each relevance feedback instance.

$$vd_+ = \max_i\{vd_i\},\ vd_- = \min_i\{vd_i\},\ td_+ = \max_x\{td_x\},\ td_- = \min_x\{td_x\}$$

Rule 3: We don't compute the rank scores of relevance feedback instances. They are fixed to current maximum (minimum) scores in the candidate set.

$$\mathcal{Q}_k(a_+) = \max_i \mathcal{Q}_k(a_i),\ \mathcal{Q}_k(a_-) = \min_i \mathcal{Q}_k(a_i),$$
$$\mathcal{Q}_k(t_+) = \max_x \mathcal{Q}_k(t_x),\ \mathcal{Q}_k(t_-) = \min_x \mathcal{Q}_k(t_x),\ 0 \le k \le h$$

With these rules, positive instances can contribute more scores in the propagation and refining process so that their related images and tags can gather more scores and be ranked higher; negative instances can only contribute fewer scores so that their related ones can gather fewer scores and be ranked lower.

4   Experimental Results

4.1   Experimental Settings

The dataset we used for this experiment is NUS-WIDE [13]. It was created by downloading images and their social tags from Flickr. It has 269,648 images and about 425,000 unique original tags. For images, it provides six types of low-level features extracted from the images, which we have introduced in section 3.2. For tags, the authors of this dataset set several rules to filter the original tag set. They delete those tags that have a frequency below a certain threshold; the low frequency threshold is set to 100. They also remove the tags that do not exist in WordNet. At the end, they counted 5,018 unique tags. We retain this filtering in our experiment for the following reasons: it reduces the noises in the tag set; and it reduces the size of candidate tag set $\mathcal{T}$ and the number of links between images and tags, which can reduce the time cost of the ranking computation.

NUS-WIDE provides image annotation ground-truth of 81 concepts for the entire dataset, but it does not appoint a query sample set or provide ground-truth for content-based image retrieval. We need to construct these components by ourselves for our experiments. In our experiments, we randomly choose 100 images as a query image set (from the entire dataset) for our evaluation. We choose 20 queries in this query image set as a training set; the other 80 queries become the testing set. Note that there is no textual information available for
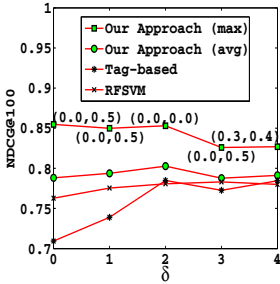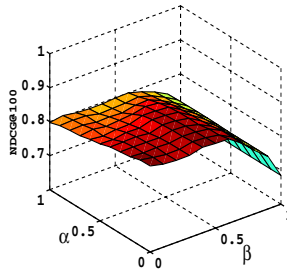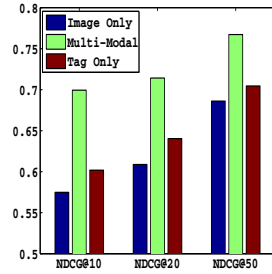
Fig. 4. δ for td_x **Fig. 5.** α and β **Fig. 6.** Multi-modal

these images when they are used as queries. For each query, we re-rank the images in top-n content-based similar image results, the cut-off size $n = 100$. For the experiments to be repeatable and comparable, we develop a rule for relevance feedback selection. For each query session, a user provides one round relevance feedback information by selecting one positive instance and one negative instance for both images and tags (four instances in total).

The images in the content-based similar image results are labeled on a scale with five levels, according to the relevance degree to the queries, by human beings. The range of levels is from 0 (irrelevant) to 4 (relevant). The evaluation metric used in our experiment is Normalized Discounted Cumulative Gain (NDCG) [14]: $NDCG@k = Z_k \sum_{j=1}^{k}((2^{r(j)} - 1)/log(1 + j))$. NDCG is an effective metric for evaluating the rank results with relevance levels. $r(j)$ is the relevance level of the image at rank j. Z_k is a normalization constant and equal to the maximum DCG value that the top-k ranked images can achieve, such that the NDCG score is equal to 1 for optimal results.

We use the training set with 20 queries for parameter selection. The proper local frequency threshold δ of td is different for different approaches. Fig.4 shows the maximal NDCG@100 value and average NDCG@100 value that our approach can reach with different δ. In this figure, e.g., if $\delta = 2$, our approach can reach the maximal NDCG@100 value when (α, β) is set to $(0.0, 0.0)$. The average NDCG@100 value is the mean for different pairs of (α, β). Fig.5 shows how we set and select suitable iteration parameters α and β in our mutual reinforcement approach. We choose their candidate values at an interval of 0.1 in the range of $[0, 1]$ and obtain 121 pairs of candidate values. We run our approach with these pairs on the training set and observe the performance. According to Fig. 4 and Fig.5, we select $\delta = 2$ and (α, β) as $(0.0, 0.0)$ for our approach. The upper limit of iteration times is set to 10 because we observe that the computation generally converge in several iterations.

4.2   Approach Comparison

We compare our approach with several other approaches as follows. The parameter selection and tuning is carried out for each approach.

Image Similarity + RF. This approach moves the positive (negative) instances to the top (bottom) rank positions.

VisualRank + RF. This approach does not use any social tag information. It is based on VisualRank [2] with modifications to make it applicable to our scenario. The iteration formula is $\mathcal{Q}_{k+1}(a_i) = (1 - \gamma) * \frac{1}{n} + \gamma * \sum_j (\mathcal{Q}_k(a_j) * \frac{s_{ij}}{\sum_x s_{xj}})$. We follow the settings in VisualRank and set the damping factor γ to 0.85. n is the size of candidate image set $\mathcal{A}$. To leverage relevance feedback information, rules in Section 3.5 are added to this approach. Although VisualRank performs well in [2], the initial image results in that work are taken from keyword-based image retrieval. In other words, VisualRank uses both visual and textual information to generate the final results. However, in our scenario, because the initial image results are content-based, only visual information can be used.

Naive Tag-Based Approach + RF. To show that our mutual reinforcement process can use social tag information more effectively, we design a naive tag-based approach that also uses social tag information for re-ranking. We compute the rank score of candidate image a_i by using the following formula: $\mathcal{Q}(a_i) = \sum_{\forall a_i : t_x \in \mathcal{T}_{a_i}} \Phi(td_x)$. To leverage relevance feedback information, rules in Section 3.5 are added to this approach. According to Fig 4, we set $\delta = 2$.

Tag-Visual-Based Approach + RF. The naive tag-based approach is based on the content-based similar search results, but it does not use visual information in the re-ranking computation. We set this approach to show that adding visual information in re-ranking formula can improve its performance, though it is still worse than our approach. The formula is $\mathcal{Q}(a_i) = (1 - w_v) * \sum_{\forall a_i : t_x \in \mathcal{T}_{a_i}} \Phi(td_x) + w_v * \sum_{a_j \in \mathcal{A}} \frac{s_{ij}}{\sum_x s_{xj}}$. We tune the weight $w_v = 0.1$ using the training set and set $\delta = 2$.

Query Point Movement (QPM). We refer the QPM approaches proposed in [4,5] which are on low-level visual feature space. We re-compute the rank scores by using visual similarity and textual distance between image RF instances and candidate images: $\mathcal{Q}(a_i) = \Phi(vd_i) + \sum_{a_r \in \mathcal{RF}_{\mathcal{A}}} (-1)^k (\frac{s_{ir}}{\sum_x s_{xr}} + d(\mathcal{T}_{a_i}, \mathcal{T}_{a_r}))$, $k = 0$ for positive instances and $k = 1$ for negative instances. The textual distance $d(\mathcal{T}_{a_i}, \mathcal{T}_{a_r})$ is the cosine distance of two tag set. δ is also set to 2.

Naive Mutual Reinforcement. The mutual reinforcement process is not particularly novel and some approaches based on it have been proposed in other fields, e.g. link analysis [16]. We set this approach to show that the introduction of the weights in the 2nd term of the iterative formulas, and the cautious selections on the visual and textual descriptors and normalization function, yield better performance than the naive one.

$$\textit{Initialization: } \mathcal{Q}'_0(a_i) = \Phi'(vd'_i), \; \mathcal{Q}'_0(t_x) = \Phi'(td'_x); \; 0 \le \alpha, \beta \le 1;$$

$$\textit{Iterations: } \begin{cases} \mathcal{Q}_{k+1}(t_x) = \alpha \Phi'(td'_x) + (1 - \alpha) \sum_{\forall a_i : t_x \in T_{a_i}} \mathcal{Q}'_k(a_i) \\ \mathcal{Q}_{k+1}(a_i) = \beta \Phi'(vd'_i) + (1 - \beta) \sum_{\forall t_x : t_x \in T_{a_i}} \mathcal{Q}'_k(t_x) \\ \mathcal{Q}'_{k+1}(t_x) = \Phi'(\mathcal{Q}_{k+1}(t_x)), \; \mathcal{Q}'_{k+1}(a_i) = \Phi'(\mathcal{Q}_{k+1}(a_i)) \end{cases}$$

$$\Phi'(\mathcal{Q}_k(t_x)) = \frac{\mathcal{Q}_k(t_x)}{\sqrt{\sum_y \mathcal{Q}_k(t_y)^2}}, vd_i' = vd_i, td_x' = \sum_{t_y \in \mathcal{T}} \frac{|t_x \cap t_y|_{\mathcal{T}}}{|t_x \cup t_y|_{\mathcal{T}}}.$$

The text descriptor here uses a tag co-occurrence measure referring to [12]. $|t_x \cap t_y|_{\mathcal{T}}$ ($|t_x \cup t_y|_{\mathcal{T}}$) means the number of images that contain both of t_x and (or) t_y. (α, β, δ) is also chosen as $(0.0, 0.0, 2)$, same with our approach.

RFSVM. We refer the approaches proposed in [9,10] which use SVM to relevance feedback for re-ranking. We use the libsvm [15] to implement this approach. In our scenario, we construct the SVM vectors of each image a_i as

$$v_i = \{vd_i, s_{i1}, \cdots, s_{in}, td_{i1}, \cdots, td_{im}\}, \ td_{ix} = \begin{cases} td_x, t_x \in T_{a_i}, \\ 0, t_x \notin T_{a_i}. \end{cases}$$

As the number of tags is not fixed for each query session, the size of the vector is not fixed: it is equal to $1 + n + m$. In each round, we use the vectors of the relevance feedback image instances as training set for the learning processing of SVM, and the vectors of all images in the candidate image set $\mathcal{A}$ as testing set. We scale down the testing data and training data in that order. Learning is performed with the *default* settings in libsvm. According to Fig 4, we set $\delta = 3$.

RFClustering. We refer the approaches proposed in [3,8] which use clustering methods for re-ranking. The steps of this approach are as follows: (1) use kmeans method to cluster all images in $\mathcal{A}$; (2) rank clusters based on image similarity and relevance feedback information (with rule 2) in clusters; (3) rank images with the fusion of image similarity and cluster rank. The vectors of RFClustering are same with that of RFSVM. We tune the number of clusters to 5 using the training set and set $\delta = 3$.

4.3   Experimental Results

We use the testing set with 80 queries for performance evaluation. Fig. 7 illustrates the results for NDCG@10, NDCG@20, NDCG@50 metrics. This figure shows that our approach performs better than all other approaches.

Fig. 6 illustrates that, compared with using only image or tag relevance feedback instances, using both image and tag instances enables better performance. Compared with the traditional single-modal relevance feedback scheme, our multi-modal scheme significantly improves the performance.
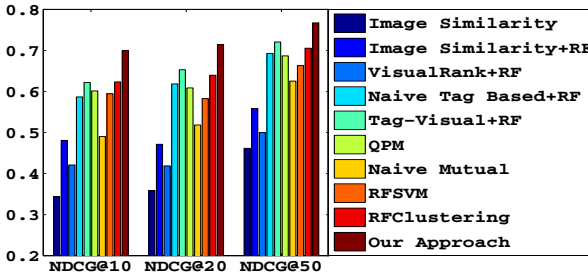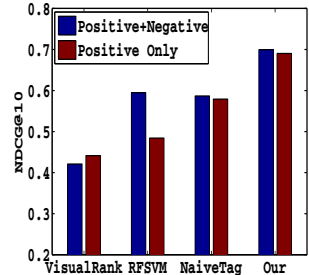

Fig. 7. Approaches Comparison

Fig. 8. RF Type

Fig. 8 indicates that our approach performs well even when only positive instances are available. This means that even if users only label several positive instances they are interested in, without labeling any negative instances, our approach still works well and performs better than other approaches.

5   Conclusions

In this paper, we successfully use social tags to improve content-based social image search results. We propose a re-ranking approach with a mutual reinforcement process using both visual and textual information, along with multi-modal relevance feedback scheme, on a image-tag relationship graph model. In future work, we will extend our work to a keyword-based image retrieval scenario.

References

1. Bischoff, K., Firan, C.S., Nejdl, W., Paiu, R.: Can all tags be used for search? In: CIKM, pp.193–202 (2008)
2. Jing, Y., Baluja, S.: VisualRank: Applying PageRank to Large-Scale Image Search. TPAMI 30(11), 1877–1890 (2008)
3. Park, G., Baek, Y., Lee, H.: Re-ranking algorithm using post-retrieval clustering for content-based image retrieval. Inf. Process. Manage. 41(2) (2005)
4. Porkaew, K., Mehrotra, S., Ortega, M.: Query reformulation for content based multimedia retrieval in MARS. In: ICMCS, pp. 747–751 (1999)
5. Porkaew, K., Chakrabarti, K., Mehrotra, S.: Query Refinement for Multimedia Similarity Retrieval in MARS. In: ACM Multimedia, pp. 235–238 (1999)
6. Yan, R., Hauptmann, A., Jin, R.: Multimedia search with pseudo-relevance feedback. In: CIVR 2003, pp. 238–247 (2003)
7. Liu, Y., Mei, T., Hua, X., Tang, J., Wu, X., Li, S.: Learning to video search rerank via pseudo preference feedback. In: ICME 2008, pp. 297–300 (2008)
8. Hsu, W., Kennedy, L., Chang, S.: Video Search Reranking via Information Bottleneck Principle. In: MM, pp. 35–44 (2006)
9. Zhang, L., Lin, F., Zhang, B.: Support vector machine learning for image retrieval. In: ICIP, pp. 721–724 (2001)
10. Chen, Y., Zhou, X., Huang, T.: One-class SVM for learning in image retrieval. In: ICIP, pp. 34–37 (2001)
11. van de Sande, K., Gevers, T., Snoek, C.: Evaluating Color Descriptors for Object and Scene Recognition. TPAMI 32(9), 1582–1596 (2010)
12. Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: WWW, pp. 327–336 (2008)
13. Chua, T., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: NUS-WIDE: A Real-World Web Image Database from National University of Singapore. In: CIVR (2009)
14. Jarvelin, K., Kekalainen, J.: Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems (TOIS) 20(4) (2001)
15. Chang, C., Lin, C.: LIBSVM: a library for support vector machines (2001) Software, `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
16. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)

Towards Real Intelligent Web Exploration

Pavel Kalinov, Abdul Sattar, and Bela Stantic

Institute for Integrated and Intelligent Systems,
Griffith University, Brisbane, Australia
pavel.kalinov@griffithuni.edu.au,
{A.Sattar,B.Stantic}@griffith.edu.au
http://iiis.griffith.edu.au/

Abstract. A significant problem of the dominant web search model is the lack of a realistic way to acquire user search context. Search engines use implicit feedback, which is extremely sparse and does not allow users to properly define what they want to know, or what they think of search results. In our proposed "web exploration engine", which we implemented as a prototype, documents have been automatically pre-classified into a large number of categories representing a hierarchy of search contexts. Users can browse this structure or search within a particular category (context) by explicitly selecting it. Keyword relevance is not global but specific to a category. The main innovation we propose is the "floating" query resulting from this feature: the original search query is re-evaluated and the importance of its features re-calculated for every context the user explores. This allows users to search or browse in a truly local (context-dependent) way with a minimum of effort on their part.

Keywords: Web Directory, Relevance Feedback, Adaptive Ranking.

1 Introduction and Motivation

Scientific research on web information retrieval systems is concentrated primarily on search engines, which are also the dominant information-finding model on the web. However, many unresolved issues have arisen from their dominance.

Search engines by design satisfy the *information locating* need only (the user knows something exists and needs to find out where it is). They do not address the complementary *information discovery* need (the user does not know something and needs to find out that it exists). This is served by web directories, which are now in decline (mainly due to not implementing machine learning).

A major issue that recently began to be recognised arises from the fact that search engines limit the number of results shown to users. With a finite number of keyword combinations related to every document, it is very probable that (given a consistent ranking algorithm) a significant number of documents will always rank below the "decision boundary" and be unreachable through keyword search [1], *even if people search for them using the correct keywords*. A different ranking algorithm might put these documents before the threshold, but users are not allowed to influence the algorithm or select a different one.

Originally, search engines were built around some basic assumptions: that every search is separate and independent; every search is generated by a short-term interest; every user means the same with the same word; users require fast answers; users cannot (or will not) explicitly supply context for their queries; advanced features should be sacrificed to simplify search interfaces. These assumptions are the basis of the current system-centric model; many features and methods are introduced to address issues arising from them, but these are only mitigation efforts while the underlying model remains the same.

Additionally, search context and the user's "background knowledge" (cognitive context), as well as *synonymy* and *polysemy* [2] are largely missing from the picture (again, with many mitigation efforts which do not change the underlying model). To optimise for speed, most search results are pre-computed and identical for all users and contexts, with minimal personalisation at retrieval time to fit certain pre-computed criteria [3]; search queries are limited to a (short and flat) list of words: users lack an expressive way to define what they search for; they cannot adjust result ranking factors, or even know what they are.

Furthermore, some exploitable features of popular search engine algorithms have had a large negative impact on the web as a whole. A whole "search engine optimisation" industry arose out of these features, employing practices such as "link farming" (creating sites for the express purpose of filling them with links to other sites in order to increase their "popularity"), "keyword stacking" (publishing automatically generated text filled with desired keywords to increase "keyword density"), "keyword bombing" (linking to sites with specific keywords in the link to improve their ranking for those words), the creation of "fast food content" (low-quality texts aimed exclusively at gaining high search engine ranking so as to place advertising on them) etc. They lead to a significant and growing proportion of web content being "digital garbage" aimed at search engines and not humans, making the information-finding task ever more difficult for users.

Web directories, if they manage to overcome some of their problems, could provide a viable alternative addressing some of these issues. For example, a large enough browsable hierarchy of topically ordered documents could allow users to reach any document with a finite number of navigational clicks, whatever its ranking is; while browsing, users would see branches they never knew existed (thus would never search for) and would discover new information; search within a category would limit the scope of their search, providing some search context.

However, web directories face even more serious problems than search engines. Some are inherent, like the difficulty users have of navigating a large tree-like structure (the Open Directory has over 700,000 categories), where they have to take a decision at every level: which branch down to follow? This is not as trivial as it seems, since it requires users to a) know exactly what they want, and b) classify it the same way as the directory maintainers do (e.g., somebody looking for dolphins under "Fish" will never find them).

Information acquisition is manual: entries are individually added by humans. This results in web directories having several orders of magnitude less data than search engines, and in whole types of web documents completely missing from

them. For example, editors may link to the home page of an online newspaper, but not to its individual articles. They also avoid documents which tend to disappear or change too often, since maintaining the integrity of already published information is another significant problem related to the limitations of the manual maintenance model, which suffers heavily from "web decay" [4].

Another problem is the difficulty of using machine learning for building a web directory. While it is essentially a text classification problem, there are some web-specific issues making the use of popular text classification algorithms impractical. The biggest of them is that the web grows and changes constantly [5]. Human perception of resources also changes, so a resource may be initially classified into a category and then re-classified to another. A "batch" text classifier would need to be retrained from scratch to reflect such changes, which is not a feasible task if it has to be done often on a real-world web scale. "Streaming data" algorithms can deal with online addition of data, but not with removal/modification of instances already processed, or their re-classification.

A further problem is that statistical text classification algorithms are based on some assumptions: feature independence, relatively small variance in basic document statistics like word count per document etc.; these assumptions are usually broken in real-world texts, but classifiers manage to overcome this by applying different normalisations. It turns out though that documents from different categories break these assumptions differently and some vital statistics vary greatly between classes (our experiments found that documents in the "Business" category of our sample of downloaded Open Directory sites had an average of 96.54 unique terms each while "News" had 235.05 [6]). As a result, a classification method that works in one category ("Sports" sites) does not work for another ("Business" sites) [7]. Every separate issue can be overcome by some heuristic, normalisation or parameter setting, but the variance of issues between categories means that different solutions have to be found for almost every category. This task is comparable in complexity to the manual classification of all instances and is probably why machine learning has never been applied to a major directory.

One way to create a hybrid between a search engine and web directory is to apply clustering of search results and present clusters to the user as search contexts [3], for which also an open-source framework exists[1]. However, this clustering is applied over the (limited) search results provided by the search engines and expressly cannot classify documents into pre-existing categories. These clusters are also typically a small number and only one level deep, i.e.: they cannot achieve a deep topical hierarchy.

An alternative approach could combine some features of search engines with some features of web directories and use the advantages of one model to overcome inherent problems of the other. For example, using an automated data collection mechanism (a web spider) would be highly beneficial to web directories, while the ability of users to browse topically ordered information will provide the "information discovery" lacking in a search engine, and the ability to narrow down search to a particular category will provide some search context.

---

[1] Carrot^2: http://project.carrot2.org/

2   Web Exploration Engine

We propose a system which is a cross between a web directory and a search engine: it collects and indexes documents using a search engine-like web spider, but then classifies them into a hierarchically ordered set of categories. These act as pre-made contexts and allow users to limit their search (when performing keyword search), but are also browsable allowing users to find information without having to supply keywords which is difficult in cases where users cannot specifically define what they are looking for (i.e. half of search cases [8]).

Users can search the directory or specific parts of it, and are able to influence result ranking by providing explicit relevance feedback which is then integrated into the research query. We have named it the "research query" as opposed to "search query" to stress the fact that it is not a flat keyword list but a complex user-adjustable vector where term weights can even be negative (if the user indicated negative preference towards some resource); the user builds this query over a period of time, and can then save it and re-use it later, allowing long-term research over a topic. The initial query is optional since the user can start by just browsing (having and empty query) and supplying relevance feedback ("this site seems like what I want", "this site is something I do not want"), which then creates a query from the feedback instead of just adding to the initial one.

This "Web Exploration Engine", as we have named it, enables users to:

- browse a directory structure (providing *information discovery*);
- search by keywords (providing *information locating*);
- limit the search within a branch of the directory tree, enabling a focus on a narrower (predefined) context;
- create and/or expand a query by supplying relevance feedback, providing explicitly specified user context;
- expand the query in a "session" manner, with small increments leading to a detailed, in-depth query;
- save a query for later re-use and expansion, enabling long-term research.

To build this structure, we use a statistical classifier trained on human-labelled examples. Its hierarchy can be arbitrarily deep, with relatively few documents per leaf node, so that every document is reachable by simple browsing. If we suppose an average of 100 documents per leaf node and 10 branches per node, users would be able to browse 100 billion documents by ten clicks on average.

2.1   Building the Directory

As opposed to the traditional model, our system [9] uses a web spider to obtain its data. The spider is semi-automated: it does not follow all links automatically but is guided by a human editor (URLs have to be approved before being downloaded). This assures a certain level of quality, e.g. - the editor may decide to not index forum postings or other user-generated content, and allows a certain focus by topic if we want to build a "vertical" (topic-specific) directory: the editor can

approve only links outgoing from relevant sites. Processing the download queue is assisted by auto-approval by manually added URL patterns ("if the URL starts with *http://www.bbc.co.uk/news/* - download it without specific approval").

We train a hierarchical structure of Multinomial Naïve Bayesian classifiers [6] on labelled URLs from the Open Directory which are classified by human editors into a tree of categories. Expanding this information in our system is assisted by heuristic pattern-based classification: we added classification rules such as "if the URL starts with *http://www.transdat.com.au/* and contains machine*.html" - add it to the Business / Construction / Equipment / Machines subcategory" (where * represents a wildcard). Documents downloaded by the spider that are not classified manually or by pattern are added to categories based on the classifier's decision; this increases content by several orders of magnitude.

We can make human labour more efficient by using the classifier scores: editors can assess and classify only "borderline" cases with low classifier score, and not waste time on clear-cut cases. Conversely, cases with too high classifier confidence can also be examined to prevent "search engine optimised" documents (which can be considered a form of spam) from taking advantage of some feature of the algorithm. Additionally, the editors' attention can be focused on any atypical documents. For example, our system calculates the average term TF-IDF of every document; our experiments showed that most outliers (documents with extreme average values) are either errors or web spam: machine-generated text used for "keyword stacking", or random gibberish to confuse spam filters ("Bayesian poisoning" [10]). These can be weeded out at a relatively low cost, as compared to a human checking every document in the directory.

For hierarchical classification we use not one multi-level classifier but a tree of separate independent classifiers; outputs from each one (in the form of lists of documents belonging to a category) are inputs for those at the lower level. Each classifier works with data from its category only and calculates static measures locally. Among other things, this means there cannot be global stop-words. Stop-words, and word weights in general, are context-specific: the usual example are words such as "the" and "and" which are too common in the English language and useless for classification. However, their frequent appearance in a text means that the text is (very probably) in English, so they are useful at the first level of our tree where we separate English-language from "other language" documents. This applies to a differing extent to all words in all contexts, so we calculate their weights separately for every context (category). Consequently, we do not use linguistic pre-processing which relies on stop-word filtering. On the plus side, using a number of independent classifiers allows us to create a distributed system to work with web-scale data, with separate servers for each category.

To develop the classification part of our prototype, we experimented [6] using labelled data from the Open Directory Project. We downloaded a sample of documents and trained on their original texts and not the ODP descriptions which tend to be very short, as well as repetitive hence not very distinguishing.

The first issue we faced was lack of sufficient labelling data, which is extremely sparse. The Open Directory contains an average of only 6.02 documents per

category. From an end-user point of view browsing such categories is impractical, and from our point of view training a classifier on only 6 instances is impossible. To address this, we "folded" categories to include all instances from descendant nodes as well (the same approach was used in other projects [11,12] based on the Yahoo! Directory, where category fragmentation is even worse).

We chose Multinomial Naïve Bayes as our main classification tool because it learns incrementally, classifies fast and generalises well. However, it has serious problems with very noisy or unbalanced data (such as web documents).

We applied several mitigation measures. The first was TF-IDF term weighting to discount frequent terms, which is standard practice in text classification. However, as already discussed, we use local values for the IDF part as it is calculated over a partial document collection (those documents belonging to a category), so the same word has different IDF values in different parts of the classification tree. We also applied word count normalisation [13] to compensate for the varying average length of documents between classes:

$$n'_{wd} = \alpha \times \frac{n_{wd}}{\sum_{w'} \sum_{d \in D_c} n_{w'd}} \tag{1}$$

where $n_{w'd}$ are l_2 normalised class-specific word counts (number of occurrences of the word w in documents d of the part of the corpus D_c belonging to class c); we took the smoothing parameter α (vector length measured in the l_1 norm) to be equal to 1, as that was shown to work well [13] and we did not want to introduce a new point of failure by experimenting with it further. This normalisation improved classification success considerably, but results were still not satisfactory as a whole. The main issue was not so much the overall classification success rate as its variance between classes: for some categories the classifier was almost perfect while in some categories it had a success rate of practically zero.

We then decided to apply a policy which proved to be effective for industrial spam filters [10]: *train on error*, meaning the classifier trains only on documents which it misclassified. Obviously, this skews word counts and prior distributions, since the classifier "sees" only a small part of the document collection (the borderline cases) and ignores most of the typical documents. It is counter-intuitive, but works in the real-world, which was also confirmed by our experiments.

Apparently, modifying the basis on which word counts and prior distributions are calculated serves to improve the situation dramatically. We then tried several unsuccessful new variations until we made a breakthrough which actually changed the fundamental workings of the algorithm. We decided to calculate class prior distributions dynamically: over a sliding window of the last 10 000 errors and not over the whole document collection. This introduces a negative feedback loop: problematic classes become over-represented in this stochastic sample (there are disproportionately more errors in them) so their prior probability is adjusted upwards to compensate for the problems (note that we do not even need to know what the problems are). Of course, a price has to be paid: increased accuracy in some classes happens at the expense of classes where the classifier was previously more accurate. Nevertheless, overall accuracy increases

and - more importantly - error variation between classes drops four times [6], meaning that the classifier is now equally reliable for all classes.

Fundamentally, we now have a different type of algorithm. It is based on Multinomial Naïve Bayes but is no longer a static batch-classifying algorithm. Instead, it is dynamic in the sense that it needs to keep classifying (and make some new errors!) in order to learn. The negative feedback loop keeps it in a dynamically stable state - in this case, minimising error variance between classes (note that it does not minimise errors themselves, just their inter-class variance, so it can never be completely error-free).

This approach leads to some important consequences that have to be noted:

– The algorithm has a learning stage: we have to keep classifying (and making errors) until it converges to a stable state. We found experimentally that this happens after four or five iterations over the whole document corpus.
– Classification order matters! If we do not randomise iterations properly and, for example, start classifying with instances of one class only it can easily get unbalanced and go into wild fluctuations that are difficult to recover from.
– After we have classified the whole document corpus, we re-initialise and keep classifying in order to stay up-to-date. Essentially, the classifier never stops.
– The classifier works in parallel with the web spider, which keeps updating the document corpus (adds new documents, deletes documents that were removed from the web and updates documents that changed), and with the editors who add documents but may also move them from one category to another. To avoid potential conflicts, the classifier works not with current data but with a snapshot of the data as it was when an iteration started.
– Since the classifier keeps re-training, word counts would potentially reach enormous values. To avoid this, we introduced weight decay applied at the start of each iteration. This means that a term which has not been encountered for some time will have its weight decreased and will be eventually deleted. We get two benefits from this: if we move a document from one category to another, the classifier "forgets" the initial classification after some time (automatic re-learning), and if a document gets removed from the collection altogether the terms associated with it are eventually deleted. This automatically deals with Bayesian poisoning, because documents containing it tend to be short-lived (comment spam or spam text injected into hacked legitimate sites which gets deleted when the site owners discover it).
– Since the method deals automatically with any imbalances, noise and other problems in the data, there are no heuristics or parameter settings to modify for its various use cases (which is a major issue for industrial spam filters [10]). We can apply it in every part of the category tree without modification.

Having this algorithm, we can build a usable large-scale web directory with relatively low human labour. It has to be stressed though that the method is not fully automatic as it inherently needs manual labelling of some data instances, as well as creating the category structure. Nevertheless, we believe it is much more viable from an economic point of view than the current dominant model.

2.2   Browsing the Directory

The directory can be browsed as a simple static hierarchical structure similar to the Open Directory. We have added some document ranking options though to improve usability. Users can browse documents alphabetically, ordered by editor's choice, by *typicality* (how typical a document is for the category - by its classifier score) or by relevance (similarity) to the current user query.

The main feature of the system however is its *Exploration* mode. Users can start exploring by submitting a query in the form of several keywords, a short text or a whole document, or they can just start browsing the directory - i.e., have an empty query; in both cases, this query can later be expanded.

The engine returns results in the form of a path through the directory tree, as well as some document listings. It treats the query as a document, passes it through the pre-processing filter of the backend classifier and converts it into a TF-IDF weighted vector (we have TF because words can have more than one occurrence in the query, and initially we use IDF values for the whole document corpus). This weighted vector then goes through the many levels of classifiers, where the IDF parts of term weights are substituted with their local values. At each level, the most probable category is returned as the answer but other categories are listed as well, in order of probability (i.e. - how well they fit the query). They are illustrated by several "most typical examples": the top N instances by order of classifier score.

Users can follow the suggested path through the directory and see listings at different depth levels. The ability to "jump" straight to a low-level category without having to manually select branches at higher levels significantly assists users from a usability point of view but also from a cognitive point of view: those who would otherwise look for dolphins under "Fish" would now learn that they are "Mammals/Marine" instead. Users can also correct the system by following an alternative path; or, even if the classifier is correct, they can still click on alternative links and explore related areas using it as a recommendation feature.

2.3   Searching in the Directory

As all search engines do, we use a document "posting list": an inverted index which is the reversed version of the document description table (instead of indexing words contained in a document, it indexes documents that contain a word). In our case though this is not a flat occurrence list with *yes* or *no* values or number of occurrences. We have a normalised weight for each word for each document: $sw_i = \dfrac{tf\text{-}idf_{wi}}{tf\text{-}idf_i}$ - the specific weight sw_i for the word in document i is the ratio between the word's TF-IDF value for that document and the average TF-IDF of all words in the document. As with every other use of IDF values in our system, these are local, calculated for the specific category.

On this basis we can compare the relative importance of words in documents and can say for example: word w_i is twice more important for document D_j than for document D_k **in category** C. We order candidate documents by these scores to rank them by relevance to the research query.

As discussed, a word is common or distinctive in some context only. The word *software* can provide a good decision boundary when splitting IT-related from other texts, but once we have texts about software only it stops being distinctive and should already be treated as a stop-word. This is why having local weights is an advantage for our system: at the top level, users receive results from the whole document collection, ordered by global relevance (just as if they search a normal search engine); when they select a specific category though, we perform local search: over a partial document collection and *ordered by local relevance*. In effect, we provide a cascading series of increasingly specialised search engines allowing focused search in pre-defined contexts. Search can be narrowed down by selecting from millions of nested contexts with only several clicks.

For technical reasons, we limit the number of search results returned to the user to 1000. Limiting the number of search results would normally produce the effect that some documents become unreachable by search. In our model however, the user can browse the directory tree into more specific categories thus a) narrowing the document collection, b) reordering search results due to the different local ranking, and/or c) conceivably reaching a node with a handful of entries only, all visible at a glance, where search will not even be needed.

As additional usability enhancements, we added two features lacking in modern search engines: the ability of users to "bookmark" a research (the initial query plus all feedback in the form of sites marked as relevant/not relevant) and later re-use it, as well as the ability to add their own bookmarks to a list of search results, whether the exploration engine considers them relevant or not.

2.4   Query Expansion, Floating Query

The initial user query can be expanded interactively by relevance feedback. In the result list users can mark documents as either relevant or not relevant; when that happens, all the (weighted) keywords of the document are added to the query. This one-click query expansion does not require the user to enter keywords - the system supplies them instead. Since some of the added documents are positive (relevant) and others are negative (not relevant) examples, the query becomes a complex object with a positive and a negative component. To balance its three parts, we use Rocchio relevance feedback weighting [14]:

$$Q = \alpha Q_u + \beta Q_p - \gamma Q_n \tag{2}$$

where the resulting query Q consists of the original user query Q_u plus the query vector Q_p from documents marked as relevant minus Q_n (non-relevant documents). Query vectors are calculated as $Q = \frac{\sum D}{n}$, where D are the document vectors and n is their cardinality. α, β and γ are weights signifying how important each part is. By default $\alpha = 1$ and $\beta = \gamma = 0.5$, i.e. the original query is as important as all the feedback, and the positive and negative components weigh equally. Users can adjust these values on a case-by-case basis though.

Unlike other query expansion methods (such as the ARCH system [12]), our query expansion works on the document and not the category level; they use all

terms from all documents in a category while we only add/subtract terms from user-selected documents. This gives full and explicit control to the user and is more precise: it provides several orders of magnitude more granularity.

The resulting *research query* is very different from a standard search engine query: a) it can be arbitrarily long, and b) it is not a flat list of words but a weighted array, where weights can be negative. The query is passed through the hierarchical classifier which takes into account all terms to calculate results. It has to be noted though that results do not need to contain all or even most of the terms - they are just those that (probably) best relate to the query. An empty query would still return results: this will be "the most probable" path through the hierarchy, which is "the most populated categories at each level" as probability estimation would in this case be based on prior probabilities only.

Further to being complex, the query also changes as the user moves around the directory structure. Keywords in the query have an initial weight, which is then multiplied by the local weight of that keyword in the category which the user is currently in. Thus, we get a *floating query* which changes with every click the user makes, adapting to the current context. An example can be seen in the table below (values are not realistic but chosen for illustration purposes only).

Table 1. Evolution of the original search query as the user moves through the English-language category into Equipment and Aerospace subcategories

	Original	English	Equipment	Aerospace
and	0.01	**0.92**	0.00	0.00
repair	**0.47**	0.69	0.15	0.06
radar	0.23	0.03	**0.89**	0.08
helicopter	0.20	0.61	0.85	**0.98**

Note how the word *and* is initially distinctive, as it is a useful indicator to distinguish between English and other languages, then becomes a stop-word (practically disappears) as all documents in the category become English-language only. In the top level of English-language sites, *repair* is a good discriminator between Equipment and other sites, but then becomes common within that context and its importance shrinks accordingly. In the Aerospace category, *helicopter* becomes important as it distinguishes different types of Aerospace equipment.

2.5   Advantages of the Exploration Engine

The web exploration model we describe has several advantages over the existing search engine and web directory models:

– Unlike PageRank [15] and its derivatives, our ranking relies on features of the document itself and not on external factors. This renders ineffective most of the arsenal of the *search engine optimisation* industry: "keyword bombing" or "link farming" become irrelevant; "keyword stacking" is counter-productive, as adding more keywords to a document dilutes the TF-IDF values for all of these keywords and it will not rank high for any of them.

Furthermore, any *optimisation* will work in one category only, since every category has its own IDF values; thus, a document cannot be "optimised" for one category without harming its ranking in all the (millions of) others. If the model becomes wide-spread, it may deter creation of "digital garbage".

– The model uses the much more expressive explicit as opposed to implicit user feedback, allowing better user control.
– The only ranking parameter the system uses (the ratio between α, β and γ in feedback weighting) is user-adjustable, so the user has full control: we make no assumptions on his behalf.
– Unlike the dominant search engine model, in our model all documents are reachable: both by simple browsing and by focused search.
– The dynamic nature of our classifier automatically takes care of changes in classification (where editors move documents between categories), as well as Bayesian poisoning and other noise.
– The independent nature of the individual classifiers in our hierarchical classification tree allows a relatively easy implementation of a distributed system.
– Some of the features of our method facilitate more effective human editing by pointing the editors to the most problematic entries requiring attention.
– Heuristic pattern-based classification adds orders of magnitude more content into categories, used also for training data for automated classification.
– Users control the search process through explicit feedback, allowing them to better specify what they search for.
– Users can develop a query over a period of time, save it and later re-use or expand it, enabling long-term research.
– Users can add their own snippets of data to search results, even though the exploration engine may lack this data or not consider it relevant.

3   Conclusion

We propose a *Web Exploration Engine* as a hybrid information-finding model. Where current systems try to guess search or user context by implicit feedback, we use pre-computed contexts from which the user can select one and then modify it. The model also allows users to expand their research into related topics, assisting *information discovery*.

Document ranking in our search results is category- (context-) specific: the same document is ranked differently against the same keywords, depending on the context in which the user is searching. The "floating query" which enables this is the main innovation of this work.

In our future work, we will add real-world scale data, since the advantages of the engine will be best visible when it contains billions of documents in millions of categories. If using manually labelled examples on this scale this proves to be prohibitively expensive, a future development could be to allow users to assist the training process by adding a collaborative filter to the user experience.

We believe that the introduction of our classification method for dynamic data, and our approach to information locating and information discovery as a whole, make the revival of web directories practical and will allow them to easily grow into *Web Exploration Engines*.

References

1. Witten, I.H., Gori, M., Numerico, T.: Web Dragons: Inside the Myths of Search Engine Technology. Morgan Kaufmann Publishers Inc., San Francisco (2006)
2. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science 41(6), 391–407 (1990)
3. Vivisimo Inc., Tagging vs. Clustering in Enterprise Search (August 2006),
   `http://vivisimo.com/html/download-tagging`
4. Bar-Yossef, Z., Broder, A.Z., Kumar, R., Tomkins, A.: Sic transit gloria telae: Towards an understanding of the web's decay. In: Feldman, S.I., Uretsky, M., Najork, M., Wills, C.E. (eds.) Proceedings of the 13th Conference on World Wide Web, WWW 2004, pp. 328–337. ACM, New York (2004)
5. Fetterly, D., Manasse, M., Najork, M., Wiener, J.: A large-scale Study of the Evolution of Web Pages. In: Proceedings of the 12th International Conference on World Wide Web, WWW 2003, pp. 669–678. ACM, New York (2003)
6. Kalinov, P., Stantic, B., Sattar, A.: Building a Dynamic Classifier for Large Text Data Collections. In: Shen, H.T., Bouguettaya, A. (eds.) Proceedings of the 21st Australasian Database Conference, ADC 2010, vol. 104, pp. 113–122. Australian Computer Society (2010)
7. Győngyi, Z., Garcia-Molina, H., Pedersen, J.: Web Content Categorization Using Link Information. tech. rep., Stanford University (2006-2007)
8. Yoshida, T., Nakamura, S., Tanaka, K.: WeBrowSearch: Toward Web Browser with Autonomous Search. In: Benatallah, B., Casati, F., Georgakopoulos, D., Bartolini, C., Sadiq, W., Godart, C. (eds.) WISE 2007. LNCS, vol. 4831, pp. 135–146. Springer, Heidelberg (2007)
9. Kalinov, P., Stantic, B., Sattar, A.: Let's Trust Users - It is Their Search. In: Huang, J.X., King, I., Raghavan, V.V., Rueger, S. (eds.) Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2010, vol. 1, pp. 176–179. IEEE Computer Society, Los Alamitos (2010)
10. Zdziarski, J.A.: Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification. No Starch Press (July 2005)
11. Xue, G.R., Xing, D., Yang, Q., Yu, Y.: Deep Classification in Large-Scale Text Hierarchies. In: Myaeng, S.-H., Oard, D.W., Sebastiani, F., Chua, T.-S., Leong, M.-K. (eds.) Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, pp. 619–626. ACM, New York (2008)
12. Sieg, A., Mobasher, B., Lytinen, S., Burke, R.: Concept Based Query Enhancement in the ARCH Search Agent. In: International Conference on Internet Computing, pp. 613–619 (2003)
13. Frank, E., Bouckaert, R.R.: Naive Bayes for Text Classification with Unbalanced Classes. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 503–510. Springer, Heidelberg (2006)
14. Rocchio, J.: Relevance Feedback in Information Retrieval. In: The SMART Retrieval System, pp. 313–323. Prentice Hall, Englewood Cliffs (1971)
15. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. tech. rep., Stanford Digital Library Technologies Project (1998)

Continuous Min-Max Distance Bounded Query in Road Networks

Yuan-Ko Huang[1], Lien-Fa Lin[1], Yu-Chi Chung[2], and I-Fang Su[3]

[1] Department of Information Communication, Kao-Yuan University,
Kaohsiung Country, Taiwan, R.O.C.
{huangyk,lienfa}@cc.kyu.edu.tw
[2] Department of Computer Science and Information Engineering,
Chang Jung Christian University, Taiwan, R.O.C.
justim@mail.cjcu.edu.tw
[3] Department of Information Management,
Fortune Institute of Technology, Taiwan, R.O.C.
ifangsu@center.fotech.edu.tw

Abstract. In recent years, the research community has introduced various methods for processing spatio-temporal queries in road networks. In this paper, we present a novel type of spatio-temporal queries, named the *continuous min-max distance bounded query* (or CM^2DBQ for short). Given a moving query object q, a minimal distance d_m, and a maximal distance d_M, a CM^2DBQ retrieves the *bounded objects* whose road distances to q are within the range $[d_m, d_M]$ at each time instant. The CM^2DBQ is indeed an important query with many real applications. We address the problem of processing the CM^2DBQ and propose two algorithms, named the *Continuous Within Query-based* (*CWQ-based*) *algorithm* and the *CM^2DBQ algorithm*, to efficiently determine the *bounded objects* at each time instant. Extensive experiments using real road network dataset demonstrate the efficiency of the proposed algorithms.

1 Introduction

With the fast advances of positioning techniques in mobile systems, spatio-temporal databases that aim at efficiently managing moving objects are becoming more powerful and hence attract more attention than ever. Many applications, such as mobile communication systems, traffic control systems, and geographical information systems, can benefit from efficient processing of spatio-temporal queries (e.g., *KNN*, range, and within queries). Early methods [1,2,3,4,5] proposed to efficiently process spatio-temporal queries focus exclusively on Euclidean spaces (i.e., the query results are determined based on the Euclidean distance between each moving object and the query object). However, in most real-world applications, the movements of objects (e.g., cabs and pedestrians) are constrained to a transportation network. As a result, the distance between two objects should be computed based on the connectivity of the network rather than the two objects' locations so that the query results obtained from performing the early methods are useless. Recently, several studies

[6,7,8,9,10] have investigated how to process the spatio-temporal queries in road networks, where the criterion for determining the query results is the shortest network distance (i.e., shortest path) between objects. However, the focus of these studies is on providing efficient algorithms to process the *KNN* and range queries over moving objects.

In this paper, we present a novel and important type of spatio-temporal queries, named the *continuous min-max distance bounded query* (or CM^2DBQ for short). Given a query object q moving in road network, a minimal distance d_m, and a maximal distance d_M, a CM^2DBQ retrieves the objects whose road distances to q are within the range $[d_m, d_M]$ at each time instant. The objects satisfying the CM^2DBQ are named the *bounded objects*. The CM^2DBQ is a useful query that can be found in many fields and application domains. A real-world example is that of a traveler who wants to know which hotels in the vicinity are better to stay. In some cases, the traveler is not interested in the hotels closer to him/her because of their high price or poor quality. In this scenario, the CM^2DBQ can be used to find the better hotels that are not far away from the traveler, by bounding them within the distance range $[d_m, d_M]$. Another real-world application is traffic control and monitoring. Consider a set of road segments that exhibit traffic congestion. In order to effectively reduce heavy traffic in these road segments, the CM^2DBQ can be issued to find the vehicles (i.e., the *bounded objects*) outside the congested area so as to prevent them entering this area.

Let us use an example in Figure 1 to illustrate the CM^2DBQ problem, where a set of objects o_1 to o_5 moves in a road network which is represented as a graph consisting of nodes and edges. In this example, each object moves steadily towards the direction indicated by the corresponding arrow. For ease of illustration, the query object q is stationary (note that this is however not required in our method). Assume that the user wants to know within a certain period of time the *bounded objects* whose road distances to q are within the range $[d_m, d_M]$ (depicted as gray lines). As shown in Figure 1(a), at time t_1 there is no object within $[d_m, d_M]$ (i.e., no *bounded object* exists). At time t_2, the distance of object o_1 to q is equal to the minimal distance d_m (as shown in Figure 1(b)). It means that object o_1's distance is greater than d_m after time t_2 so that o_1 is a *bounded object*. As the distance of object o_2 to q is equal to the maximal distance d_M at time t_3 (as shown in Figure 1(c)), o_2 also becomes a *bounded object* after t_3 because its distance is less than d_M. Finally, the query result consists of tuples $< [t_1, t_2], \{null\} >$, $< [t_2, t_3], \{o_1\} >$, $< [t_3, t_4], \{o_1, o_2\} >$, ..., where each tuple $< [t_s, t_e], S_{BOs} >$ represents that objects in the set S_{BOs} are the *bounded objects* within time interval $[t_s, t_e]$.

To efficiently process CM^2DBQ over moving objects in road networks, we first address the problem of how to significantly reduce the overhead of representing the road distance between moving objects at each time instant. Although the road distance can be computed based on the Dijkstra's algorithm [11] or the A* algorithm [12] that have been shown to be simple and efficient for computing the road distance between stationary objects, recomputing the road distance whenever objects change locations would incur extremely high computational cost
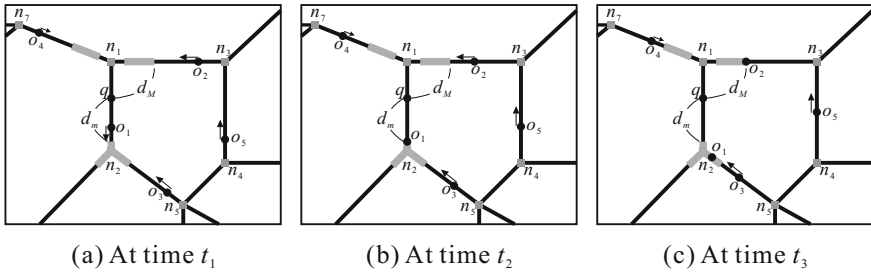
(a) At time t_1 (b) At time t_2 (c) At time t_3

Fig. 1. An example of CM^2DBQ

which makes the idea infeasible, especially for mobile environments in which objects move continuously. In order to greatly reduce the recomputation cost, we design a technique using the information about the relative speed of moving objects and the shortest path between network nodes. By exploiting this technique, the network distance between two objects at each time instant is simply represented as a linear function of time, and thus can be easily computed. Another major problem we need to tackle is to avoid repetitively processing CM^2DBQ at each time instant. Let us consider again the example in Figure 1, where S_{BOs} changes from $\{null\}$ to $\{o_1\}$ at time t_2 and then from $\{o_1\}$ to $\{o_1, o_2\}$ at time t_3. We term these time points at which S_{BOs} changes from one to another (e.g., t_2 and t_3) the *BOs-changing time points*. An important characteristic is that the *bounded objects* in between two consecutive BOs-changing time points remain the same. Based on this characteristic, the problem of performing repetitive queries can be greatly reduced to finding the BOs-changing time points and their corresponding *bounded objects*. In this paper, we design two methods combined with the proposed distance model to determine the BOs-changing time points with corresponding *bounded objects*.

To sum up, the major contributions of this paper are as follows.

- We design appropriate data structures to represent a road network consisting of edges and nodes, and to store information of data objects in network.
- A distance model is used for representing the network distance between each moving object and the query object at each time instant.
- We propose two algorithms, named the *Continuous Within Query-based* (*CWQ-based*) *algorithm* and the *CM²DBQ algorithm*, to efficiently determine the *bounded objects* at each time instant.
- A comprehensive set of experiments is conducted. The performance results manifest the efficiency and the usefulness of the proposed approaches.

The rest of this paper is organized as follows. Section 2 describes the data structures used for representing road networks and the road distance model. The *CWQ-based* algorithm and the *CM²DBQ* algorithm are presented in Section 3 and Section 4, respectively. Section 5 shows extensive experiments on the performance of our approaches. Section 6 concludes the paper with directions on future work.

2   Data Structures and Distance Model

The data structures and the distance model provide foundations for representing the road network and calculating the road distance between objects, respectively. They are also used in [13] and [14] for processing continuous K-nearest neighbor query and continuous within query, respectively. However, to make this paper self-contained, we briefly present the data structures and the key idea of the distance model here. The reader may refer to [13] for the details.

In our system, each object moves continuously in a road network, which is represented as an undirected weighted graph consisting of a set of nodes and edges. Information about edges and nodes of the network and the moving objects is stored in three tables, respectively. The first is the *edge table* T_{edge} storing for each edge e_i: (1) its start node n_s and end node n_e (where $n_s < n_e$), (2) its length *len* (i.e., the distance between n_s and n_e), (3) the maximum speed limit s_{max}, and (4) a set S_{obj} of moving objects currently on e_i. The second one is the *node table* T_{node}, that maintains for each node n_i the set S_{adj} of edges connecting n_i. The last one is the *object table* T_{obj} maintaining the information of each moving object. T_{obj} stores for each object o_i: (1) the edge e_j containing it, (2) the update time t_u, (3) the distance *dist* between o_i and the start node n_s of edge e_j (i.e., $e_j.n_s$) at t_u, and (4) its moving speed s. Fig. 2(a) is an example of road network and the detailed information of the tables T_{edge}, T_{node}, and T_{obj} is shown in Fig. 2(b).

e_i	n_s	n_e	*len*	s_{max}	S_{obj}
e_1	n_1	n_2	22	2	$\{q,o_1\}$
e_2	n_2	n_5	22	2	$\{o_3\}$
e_3	n_1	n_3	24	1	$\{o_2\}$
e_4	n_4	n_5	14	1	$\{null\}$

T_{node}

n_i	S_{adj}
n_1	$\{e_1,e_3,e_7\}$
n_2	$\{e_1,e_2,e_6\}$
n_3	$\{e_3,e_5,e_9\}$
n_4	$\{e_4,e_5,e_8\}$

T_{obj}

o_i	e_j	t_u	*dist*	s
q	e_1	0	10	-2
o_1	e_1	0	18	1
o_2	e_3	0	2	-1
o_3	e_2	0	16	2

(a) A road network   (b) Data structures
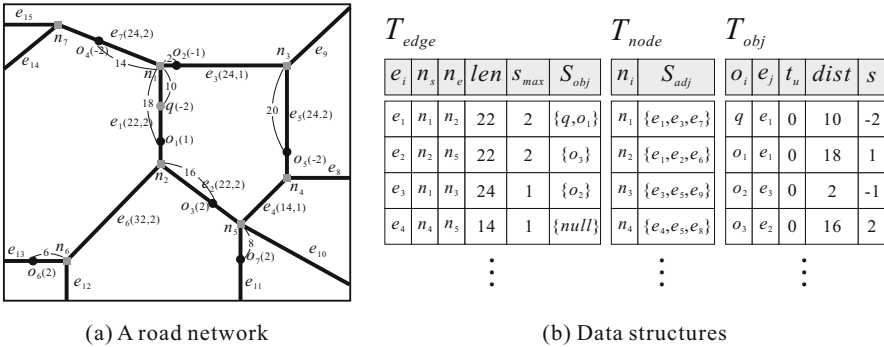
Fig. 2. Representation of road network

The road distance between each object and the query object in road networks is the primary criterion for determining the *bounded objects*. To reduce the high computation cost of the road distance, we design a technique using the information about the relative speed of objects and the shortest path between network nodes. Let the distance along the shortest path between two stationary nodes n_i and n_j on the road network be $|SP_{n_i,n_j}|$. We choose to precompute $|SP_{n_i,n_j}|$ between any pairs of network nodes and then use them to speed up the calculation of the road distance between moving objects at each time instant.

Given a query object q and a moving object o, the calculation of the network distance between q and o at any time t involves two possible cases according to whether they are moving on the same edge or not. In the first case that q and o lie on the same edge e, the network distance between q and o at time t, denoted as $ND_{q,o}(t)$, is determined as follows: $ND_{q,o}(t) = |(q.dist + q.s \times (t - q.t_u)) - (o.dist + o.s \times (t - o.t_u))|$, where $q.s$ and $q.t_u$ are the moving speed and the update time of q respectively, and $q.dist$ is the distance between q and the start node of edge e at time $q.t_u$. Similarly, $o.s$, $o.t_u$, and $o.dist$ are the information about object o. The second case is that objects q and o move on the different edges e_i and e_j, respectively. In this case, the network distance $ND_{q,o}(t)$ between q and o at time t would be equal to the sum of the network distance from q to a node n_i, the network distance from o to a node n_j, and the distance along the shortest path between the two nodes n_i and n_j (i.e., $|SP_{n_i,n_j}|$), where node n_i (or n_j) is either a start node n_s or an end node n_e of edge e_i (or e_j). By using the four distances $|SP_{n_s,n_s}|$, $|SP_{n_s,n_e}|$, $|SP_{n_e,n_s}|$, and $|SP_{n_e,n_e}|$, the network distance $ND_{q,o}(t)$ can be represented as $\min\big(ND_1(t), ND_2(t), ND_3(t), ND_4(t)\big)$, where (1) $ND_1(t) = q.dist + q.s \times (t - q.t_u) + |SP_{n_s,n_s}| + o.dist + o.s \times (t - o.t_u)$, (2) $ND_2(t) = q.dist + q.s \times (t - q.t_u) + |SP_{n_s,n_e}| + e_j.len - (o.dist + o.s \times (t - o.t_u))$, (3) $ND_3(t) = e_i.len - (q.dist + q.s \times (t - q.t_u)) + |SP_{n_e,n_s}| + o.dist + o.s \times (t - o.t_u)$, and (4) $ND_4(t) = e_i.len - (q.dist + q.s \times (t - q.t_u)) + |SP_{n_e,n_e}| + e_j.len - (o.dist + o.s \times (t - o.t_u))$. By exploiting the formulas mentioned above, the distance between two moving objects in the road network is simply represented as a linear function of time, and thus the performance of computing the road distance can be significantly improved.

3   *CWQ-Based* Algorithm

In this section, we develop a *CWQ-based* algorithm to process the CM^2DBQ, by utilizing the result of the continuous within query (CWQ) presented in [14]. The CWQ is a special case of the CM^2DBQ, which can be used to find the objects whose road distances to the query object are less than or equal to a user-given distance at each time instant. These objects are named the *within objects*. The output of the CWQ consists of tuples $< [t_1, t_2], S_1 >$, $< [t_2, t_3], S_2 >$, ..., where each tuple $< [t_i, t_{i+1}], S_i >$ represents that objects in the set S_i are the *within objects* between the time interval $[t_i, t_{i+1}]$.

The main idea of the *CWQ-based* algorithm is to perform the CWQ method proposed in [14] twice so as to obtain two sets, denoted as S^m and S^M, of the *within objects* whose road distances are less than or equal to the minimal distance d_m and the maximal distance d_M, respectively. Consider a *within object* o that is in S^M but not in S^m (i.e., $o \in S^M$ but $o \notin S^m$). As object $o \in S^M$, its road distance to the query object must be less than or equal to d_M. On the other hand, $o \notin S^m$ indicates that the road distance of object o is greater than d_m. From this, we observe that if an object $o \in S^M - S^m$, the road distance of object o is within the range $[d_m, d_M]$, and thus o is a *bounded object*. The *CWQ-based* algorithm is designed based on this observation, and the detailed steps of the *CWQ-based* algorithm are described as follows.

1. Finding the *within objects* in S^M at each time instant: performing the *CWQ* method in [14] to obtain the tuples $\sum_{i=1}^{\alpha} < [t_i, t_{i+1}], S_i^M >$.
2. Finding the *within objects* in S^m at each time instant: executing the *CWQ* method to derive the tuples $\sum_{j=1}^{\beta} < [t_j, t_{j+1}], S_j^m >$.
3. Finding the *bounded objects* in $S^M - S^m$ at each time instant: using the time points t_i and t_j (where $1 \leq i \leq \alpha$ and $1 \leq j \leq \beta$) to divide the time interval into the subintervals $\sum_{k=1}^{\gamma} [t_k, t_{k+1}]$. For each subinterval $[t_k, t_{k+1}]$, the two subintervals, $[t_i, t_{i+1}]$ and $[t_j, t_{j+1}]$, overlapping with $[t_k, t_{k+1}]$ are first determined. Then, the corresponding *bounded objects* for $[t_k, t_{k+1}]$ can be obtained by performing the set operation $S_i^M - S_j^m$.

Although the *CWQ-based* algorithm can be used to find the *bounded objects* at each time instant, it cannot achieve good performance because of the following two limitations: (1) some of the three tables T_{edge}, T_{node}, and T_{obj} are repeatedly accessed when the sets S^M and S^m are determined, and (2) the distances of the *within objects* for S_m have to be computed in finding S^M and S^m (however, these objects cannot actually be the *bounded objects* so that the unnecessary distance computations should be avoided as much as possible). To overcome the above limitations, in the next section we further propose a *CM²DBQ* algorithm to efficiently determine the *bounded objects*.

4   *CM²DBQ* Algorithm

To achieve the goal of finding the *bounded objects* of the moving query object at each time instant, the process of the *CM²DBQ* algorithm is to divide the time interval into disjoint subintervals, and these subintervals are considered sequentially in finding the *bounded objects*. As the road distances $ND_{q,o}(t)$ between all objects and the query object are required to be updated once the query object reaches an intersection (i.e., a node) of the road network, we choose to divide the time interval into subintervals by the time points at which the query object reaches a network node. That is, each subinterval refers to the time period during which the query object moves on the same edge. As a result, each object's $ND_{q,o}(t)$ remains valid within a subinterval unless it reaches a network node.

For each subinterval $[t_i, t_j]$, the procedure of determining the *bounded objects* consists of three phases. The first phase is the *inner pruning phase*, which efficiently prunes the objects whose road distances to the query object are less than the minimal distance d_m within the entire subinterval $[t_i, t_j]$. The second phase is the *outer pruning phase*, that filters out the objects whose distances to the query object are greater than the maximal distance d_M within the subinterval $[t_i, t_j]$. The above two phases are able to guarantee that a pruned object is impossible to be the *bounded object* within the subinterval $[t_i, t_j]$ regardless of whether it reaches the network node. The third phase is the *bounded objects determining phase*, which examines whether the candidates are the *bounded objects* or not. In the following, we present the three phases in details.

4.1   Inner Pruning Phase

The main idea of the *inner pruning phase* is to find a distance, denoted as d_{inner}, to ensure that if at time t_i an object o whose road distance to $q(t_i)$ or $q(t_j)$ (i.e., q's locations at time t_i or t_j) is less than d_{inner}, then o is impossible to be the *bounded object* within the subinterval $[t_i, t_j]$. The distance d_{inner} can be represented as $(d_m - d_q - d_{add})$, where d_m is the minimal distance, d_q is the moving distance of q within $[t_i, t_j]$, and d_{add} is an additional distance (which will be explained later).

Let us consider Fig. 3 to illustrate the determination of the distance d_{inner}. As shown in Fig. 3(a), eight objects (including the query object q) are moving in a road network. In this figure, each object (edge) label is followed by its corresponding speed (length and maximum speed limit) enclosed in parentheses. Assume that a CM^2DBQ is issued to find the *bounded objects* whose road distances to q are within $[20, 30]$ (that is, $d_m = 20$ and $d_M = 30$) between the subinterval $[0, 5]$ as the query object q reaches the node n_1 at time 5. As we can see from Fig. 3(b), the distance of an object o to $q(0)$ or $q(5)$ is greater than or equal to 20 (i.e., d_m) only when there exists at least one time instant $t \in [0, 5]$ such that o's location is outside the region R that starts at $q(0)$ and $q(5)$ and ends at the vertical marks with distance 15 (i.e., $d_m - d_q$). Conversely, if object o is inside the region R at any time $t \in [0, 5]$, its distance to q is always less than d_m (i.e., o cannot be a *bounded object*).

The additional distance d_{add} is used to determine whether an object is inside the region R within the entire subinterval $[t_i, t_j]$. Let the set of the boundaries of the region R be S_b (e.g., the vertical marks in Fig. 3(b)). If an object inside the region R moves with the maximum speed limit of the edge containing it within $[t_i, t_j]$, but still cannot reach any of the boundaries in S_b, then it is impossible to be the *bounded object*. Motivated by this, the distance d_{add} can be set to $\max\{d_b \mid b \in S_b\}$, where d_b refers to the maximum moving distance of an object inside the region R. Continuing the example in Fig. 3(b), the distance d_{add} is computed as $1 \times 5 = 5$. With the distances d_m, d_q and d_{add}, the distance d_{inner} can be estimated as $20 - 5 - 5 = 10$. Therefore, object o_1 inside the region starting at $q(0)$ and $q(5)$ and ending at the cross marks with distance 10 is pruned. Finally, six objects o_2 to o_7 are kept and will be further verified in the *outer pruning phase*.

4.2   Outer Pruning Phase

Similar to the *inner pruning phase*, in the *outer pruning phase* a distance, denoted as d_{outer}, is used to prune the non-qualifying objects. If at time t_i the distances of object o to $q(t_i)$ and $q(t_j)$ are greater than d_{outer}, then it cannot be the *bounded object*. The distance d_{outer} can be set to $(d_M + d_{add})$, where d_M refers to the maximal distance and the estimation of d_{add} is the same as that in Section 4.1. As shown in Fig. 3(c), the objects o_6 and o_7 that are outside the region starting at $q(0)$ and $q(5)$ and ending at the cross marks with distance 35 (i.e. $d_M + d_{add}$) can be pruned, because their distances to q at any time
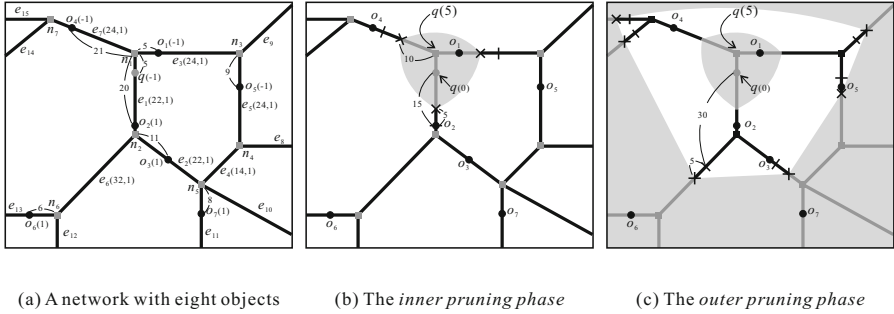
(a) A network with eight objects   (b) The *inner pruning phase*   (c) The *outer pruning phase*

Fig. 3. Example of the *inner* and *outer pruning phases*

$t \in [0, 5]$ must be greater than d_M. Only four objects o_2 to o_5 are considered in the *bounded objects determining phase*.

4.3   Bounded Objects Determining Phase

In the *bounded objects determining phase*, the interval $[t_i, t_j]$ is further divided into n disjoint subintervals $[t_{s_1}, t_{e_1}]$, $[t_{s_2}, t_{e_2}]$, ..., $[t_{s_n}, t_{e_n}]$ by the time points at which the candidates (obtained after executing the *inner pruning phase* and the *outer pruning phase*) reach the network nodes, because their $ND_{q,o}(t)$ would be changed at these time points. The goal of the *bounded objects determining phase* is to determine the BOs-changing time points within each subinterval $[t_{s_m}, t_{e_m}]$ (for $1 \le m \le n$) so that the *bounded objects* within two consecutive BOs-changing time points remain the same, and find the corresponding *bounded objects* at each BOs-changing time point.

We observe that a BOs-changing time point occurs only when one of the following four conditions holds: (1) the road distance of a *bounded object* is less than the minimal distance d_m, (2) the road distance of a *bounded object* is greater than the maximal distance d_M, (3) the road distance of an object is less than or equal to d_M, and (4) the road distance of an object is greater than or equal to d_m. Based on this observation, the time points t_m and t_M of each candidate o at which $ND_{q,o}(t_m) = d_m$ and $ND_{q,o}(t_M) = d_M$ are determined, respectively. Then, all the time points are inserted into a queue Q and sorted in ascending order. During the course of finding the BOs-changing time points, the *bounded objects* at time t_{s_m} are initially kept in a set S_{BOs} and object o with the smallest time point t in Q would be considered first. If object $o \in S_{BOs}$ and time t corresponds to t_m (or t_M), then after time t object o is removed from S_{BOs} as its distance to the query object is less (or greater) than d_m (or d_M). If object $o \notin S_{BOs}$ and time t corresponds to t_m (or t_M), then after time t object o is added into S_{BOs} because its distance is greater (or less) than or equal to d_m (or d_M). The above process proceeds until Q is empty.

Fig. 4 continues the previous example of finding the *bounded objects* within the time interval $[0, 5]$ in Fig. 3. As object o_2 reaches the nodes at time 2, the

time interval $[0,5]$ is divided into two subintervals $[0,2]$ and $[2,5]$. For the first interval $[0,2]$, the set S_{BOs} at time 0 is first determined and set as $\{o_3, o_4\}$. At time 1, the road distance of object $o_3 \in S_{BOs}$ is equal to 30 (i.e., d_M). It means that the time point 1 is a BOs-changing time point and S_{BOs} changes from $\{o_3, o_4\}$ to $\{o_4\}$. For the second interval $[2,5]$, the new $ND_{q,o}(t)$ of object o_2 will be computed at time 2 at which o_2 reaches the road node. At time 2.5, object o_2's distance is equal to 20 (i.e., d_m), so that the time point 2.5 is a BOs-changing time point and the corresponding $S_{BOs} = \{o_2, o_4\}$. At time 3, object o_4 needs to be removed from S_{BOs} because its distance is less than 20, and thus S_{BOs} changes from $\{o_2, o_4\}$ to $\{o_2\}$. Since the distance of object o_5 is equal to 30 at time 4 and $o_5 \notin S_{BOs}$, o_5 will be added into S_{BOs} (i.e., $S_{BOs} = \{o_2, o_5\}$). Finally, the query result consists of tuples $< [0,1], \{o_3, o_4\} >$, $< [1, 2.5], \{o_4\} >$, $< [2.5, 3], \{o_2, o_4\} >$, $< [3, 4], \{o_2\} >$, and $< [4, 5], \{o_2, o_5\} >$.

(a) subinterval $[0, 2]$ (b) subinterval $[2, 5]$
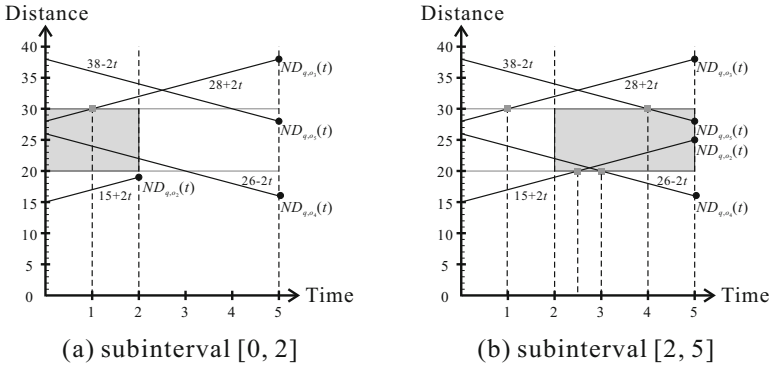
Fig. 4. Example of the *bounded objects determining phase*

5   Performance Evaluation

All the experiments are performed on a PC with AMD 1.61 GHz CPU and 2GB RAM. The algorithm is implemented in JAVA 2 (j2sdk-1.4.0.01). We use a road map of Oldenburg (a city in Germany) from Tiger/Line [15] and generate 100K objects using the generator proposed in [16]. The moving speed of each object is uniformly distributed in between 0 and 20 m/sec. When an object o reaches node n in a network, the next edge on which o moves is randomly chosen from the edges connecting n. In the experimental space, we also generate 30 query objects whose speeds are in the same range as the moving objects mentioned above. Similarly, the next edge that the query object moves on is randomly selected once it reaches a network node. The default values of the distances d_m and d_M are set to 1% and 1.5% of the entire space, respectively, and the default length of query path is equal to 100 time units. In our experiments, we compare the CM^2DBQ algorithm to the *CWQ-based* algorithm in terms of CPU cost. Three experiments are conducted to investigate the effects of three important factors on the performance of processing the CM^2DBQ. These important factors

are *the number of objects*, *the length of query interval*, and *the size of query range* (i.e., the range of $[d_m, d_M]$).

Fig. 5(a) studies the effect of the number of objects on the performance of the CM^2DBQ and the *CWQ-based* algorithms. In this experiment, we vary the number of objects from 10K to 200K and measure the CPU cost for the proposed algorithms. As we can see from the experimental result, the performance gap between the CM^2DBQ algorithm and the *CWQ-based* algorithm increases with the increasing number of objects. This is mainly because in the *CWQ-based* algorithm the objects whose distances are less than d_m have to be considered, whereas in the CM^2DBQ algorithm the unnecessary distance computation of these objects can be effectively avoided by executing the *inner pruning phase*. The increasing number of objects results in a fact that there are more objects whose distances are within the distance range d_m, and thus the improvement of the CM^2DBQ algorithm is more significant.

Fig. 5(b) evaluates the CPU cost for the CM^2DBQ and the *CWQ-based* algorithms under various lengths of query interval (varying from 10 to 100 time units). The experimental result shows that for the short query interval (e.g., 10 and 30), the performance of the CM^2DBQ algorithm is as good as that of the *CWQ-based* algorithm. However, for the long query interval, the CM^2DBQ algorithm outperforms the *CWQ-based* algorithm by a factor of 2 to 4 in terms of the CPU time. The improvement is due to the fact that the CM^2DBQ algorithm retrieves only the objects whose road distances are within $[d_m, d_M]$. However, in the *CWQ-based* algorithm the objects would be retrieved once their road distances are less than or equal to d_m or d_M.
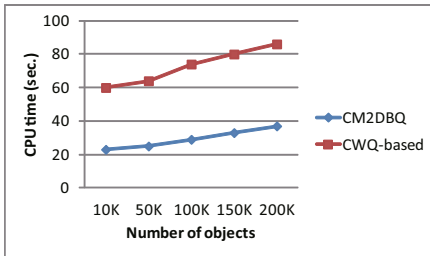
(a) Varying number of objects   (b) Varying length of query interval
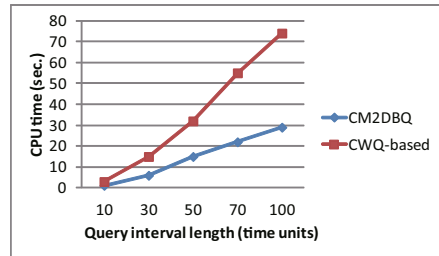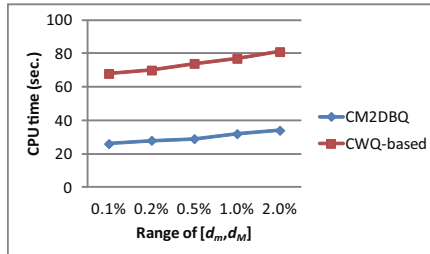
(c) Varying size of query range

Fig. 5. CPU cost

Fig. 5(c) shows the CPU time for the CM^2DBQ and the *CWQ-based* algorithms, as a function of $[d_m, d_M]$ range (varying from 0.1% to 2%). When the range of $[d_m, d_M]$ increases, the CPU overhead for both algorithms increases slightly because a larger $[d_m, d_M]$ range will result in more *bounded objects* so that more distance computations of objects are required. The experimental result demonstrates that the CM^2DBQ algorithm outperforms its competitor significantly in all cases, which confirms again that applying the *inner* and *outer pruning phases* to prune the non-qualifying objects can greatly improve the performance of processing the CM^2DBQ.

6   Conclusions

This paper focused on processing a novel and important query, the CM^2DBQ, over moving objects in road networks. Several data structures were designed to manage the information of data objects in a road network, and a technique was used to represent the road distance between objects as a linear function of time. The *CWQ-based* algorithm and the CM^2DBQ algorithm were proposed to efficiently determine the *bounded objects* at each time instant. We conducted extensive experiments to demonstrate the effectiveness and the efficiency of the proposed approaches. One important extension of this work is to address the issue of how to process the CM^2DBQ when objects change their moving speeds. A further extension is how to make proper use of data structures so as to enhance the query performance.

References

1. Huang, Y.-K., Chen, C.-C., Lee, C.: Continuous k-nearest neighbor query for moving objects with uncertain velocity. GeoInformatica 13(1), 1–25 (2009)
2. Iwerks, G., Samet, H., Smith, K.: Continuous k-nearest neighbor queries for continuously moving points with updates. In: Proceedings of the International Conference on Very Large Data Bases, Berlin, Germany, September 9-12 (2003)
3. Tao, Y., Papadias, D.: Time parameterized queries in spatio-temporal databases. In: Proceedings of the ACM SIGMOD, Madison, Wisconsin (2002)
4. Tao, Y., Papadias, D., Shen, Q.: Continuous nearest neighbor search. In: Proceedings of the International Conference on Very Large Data Bases, Hong Kong, China, August 20-23 (2002)
5. Xiong, X., Mokbel, M.F., Aref, W.G.: Sea-cnn: Scalable processing of continuous k-nearest neighbor queries in spatio-temporal databases. In: Proceedings of the International Conference on Data Engineering (2005)
6. Cho, H.-J., Chung, C.-W.: An efficient and scalable approach to cnn queries in a road network. In: Proceedings of the International Conference on Very Large Data Bases, Trondheim, Norway (2005)

7. Hu, H., Lee, D.-L., Xu, J.: Fast Nearest Neighbor Search on Road Networks. In: Ioannidis, Y., Scholl, M.H., Schmidt, J.W., Matthes, F., Hatzopoulos, M., Böhm, K., Kemper, A., Grust, T., Böhm, C. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 186–203. Springer, Heidelberg (2006)

8. Jensen, C.S., Kolar, J., Pedersen, T.B., Timko, I.: Nearest neighbor queries in road networks. In: Proceedings of the ACM GIS, New Orleans, Louisiana, USA, November 7-8 (2003)

9. Mouratidis, K., Yiu, M.L., Papadias, D., Mamoulis, N.: Continuous nearest neighbor monitoring in road networks. In: Proceedings of the International Conference on Very Large Data Bases, Seoul, Korea, September 12-15 (2006)

10. Papadias, D., Zhang, J., Mamoulis, N., Tao, Y.: Query processing in spatial network databases. In: Proceedings of the International Conference on Very Large Data Bases, Berlin, Germany, September 9-12 (2003)

11. Dijkstra, E.W.: A note on two problems in connection with graphs. Numeriche Mathematik 1, 269–271 (1959)

12. Kung, R., Hanson, E., Ioannidis, Y., Sellis, T., Shapiro, L., Stonebraker, M.: Heuristic search in data base system. In: Proceedings of the International Workshop on Expert Database Systems (1986)

13. Huang, Y.-K., Chen, Z.-W., Lee, C.: Continuous K-Nearest Neighbor Query over Moving Objects in Road Networks. In: Li, Q., Feng, L., Pei, J., Wang, S.X., Zhou, X., Zhu, Q.-M. (eds.) APWeb/WAIM 2009. LNCS, vol. 5446, pp. 27–38. Springer, Heidelberg (2009)

14. Huang, Y.-K., Lin, L.-F.: Continuous within query in road networks. In: IWCMC (2011)

15. http://www.rtreeportal.org/

16. Brinkhoff, T.: A framework for generating network-based moving objects. GeoInformatica 6(2), 153–180 (2002)

Improve Top-K Recommendation by Extending Review Analysis

Qing Zhu[1,2], Zhe Xing[1,2], and Jingfan Liang[1,2]

[1] School of Information, Renmin University of China, Beijing 100872, China
[2] Key Laboratory for Data Engineering and Knowledge Engineering MOE,
Renmin University of China, Beijing 100872, China
`zq@ruc.edu.cn`

Abstract. The Web has become the popular place for people to purchase product and acquire services, so collaborative filtering is one of the most important algorithms applied in e-commerce recommendation systems. Unfortunately, it is widely recognized that the traditional recommendation methods are inefficient when the user rating data is extremely sparse. In order to overcome the limitations, good recommendation tools are needed to help Web customers determine the products and satisfaction services. In this paper, we propose a multi-dimensional adaptive recommendation algorithm by extending opinion analysis to improve top-k recommendation. In the first step, the novel algorithm that uses extened opinion analysis, creatively combines three dimensional recommendation models: user-based, item-based and opinion-based collaborative filtering. It successfully integrates opinion mining technology with collaborative filtering algorithm. In the second step, we configured the dynamic measurement would help us determine the weight of three dimensions: user-based, item-based and opinion-based analysis, and hence get the final prediction result. The experimental results show that multi-dimensional recommendation can effectively alleviate the dataset sparsity problem and achieve better prediction accuracy compared to other traditional collaborative recommendation algorithms.

Keywords: Top-K recommendation, opinion mining, collaborative filtering, similarity criterion.

1   Introduction

With the popularity of Internet, e-commerce continues to expand the size, rapid growth of the number and types of goods, e-commerce sites need to provide personalized decision support and information services, namely providing better product recommendations services. Such recommendations can help to improve the conversion rate by helping the customer to more effectively locate products she/he wants to buy faster. Collaborative filtering (CF) is one of the most successful and widely used technologies in personalization and recommender systems.

Due to the traditional algorithms inadequacy for collaborative recommendation, in order to improve the quality of the recommendation system, many researchers have proposed their own solution. The literature[3,4] proposed the singularity value decomposition (SVD) to reduce the dimension of the space program, so that user has scored each item to project in lower-dimensional space. But the dimension reduction will lead to loss of information, it is difficult to guarantee the project effect of dimensionality reduction under the high dimensional space. [5].[6]proposed a item-based score prediction of collaborative filtering technology, which can supply the user-item score matrix to alleviate data sparseness problem at a certain extent by predicting users score projects that are not scored. However, this method still affect the quality of the recommended [1] in the calculation of similarity, because the algorithm uses the traditional method of calculating similarity.

Top-k recommendation is one of the important tasks of collaborative recommenders, but this approach does not work accurately for cold start users that have rated only a very small number of items. In this paper, we propose novel exploiting methods of multi-dimensional adaptive recommendation algorithm by extending opinion analysis to improve the quality of top-K recommendation.

The main contributions of this paper are:

- We proposed the novel algorithm that uses extended opinion analysis, creatively combines three dimensional recommendation models: user-based, item-based and opinion-based collaborative filtering.
- The method is successfully implemented by integrating opinion mining technology with collaborative filtering algorithm. Meanwhile, we configured the dynamic measurement which can determine the weight of three dimensions: user-based, item-based and opinion-based analysis, and hence get the final prediction result.
- Our experiment evaluation show that multi-dimensional recommendation can effectively alleviate the dataset sparsity problem and achieve better prediction accuracy compared to other traditional collaborative recommendation algorithms, in particular for cold start users.

Section 2 discussed related work. Section 3 details the proposed traditional collaborative algorithm. Section 4 details the proposed three dimensional recommendation models algorithm. Section 5 experimentally verified algorithm performance from multiple aspect. Finally, in Section 6 gives the summary.

2   Related Work

The literature [8] proposed the calculating method of opinion similarity vector by transfering the user emotional tendency from user review analysis. But it only determine the user's preferences through user review content analysis to understand which users pay more attention attributes of the project, and provide users with a

better recommen. [5] solve the item score data sparse matrix problems of the user-item score matrix on items not score to supplement, by predicting user ratings, has emerged in recent years. Some of the more novel research ideas: The [6]algorithm can solve part of the data sparse and cold start problems, by combine trust reasoning model with collaborative filtering recommendation algorithms. [2] proposed an uncertain neighbor collaborative filtering recommendation algorithms (UNCF, Uncertain Neighbors' Collaborative Filtering), and determined dynamic neighbors group and trust sub-group, and a combination of user-based and item-based two recommended to produce the final result recommended.These papers consider review on the network content with a strong subjective and emotional, reflecting the views of commentators, attitude and position, which has great value in use, by using text mining techniques to process these comments to obtain useful advice and knowledge to provide additional information for the recommended system. So we provide a combination of user-based, project-based and reviews-based three recommended, through considering the three dimensions of information to produce the final recommendation.

3   Traditional Collaborative Recommendation

Traditional collaborative algorithm is based on the user (user-based) and item (item-based). User-based collaborative recommendation is a representation of an active user's preferences with the historical records of past users, through the user's nearest neighbors to produce the final recommendations. Item-based collaborative recommendation algorithm is that users find k similar items rated by different users in some similar way. Then, for a target item, predictions can be generated by taking a weighted average of the target user's item scoring on these neighbor items.

Definition 1 (user-item scoring matrix). The so-called Item, that user has a certain demand for products or services, it maybe movies, merchandise, music and so on. User-item scoring matrix is used in conjunction with to create a combined similarity measure and generate predictions. In the recommended system, there is a set of s users U={ U_1 , U_2 , …,U_s }, and a collection of t items M={ M_1 , M_2 , …, M_s }, Each user can evaluate item, and have items score. The user-item scoring matrix R is showed with s × t as follow:

	M_1	…	M_j	…	M_t
U_1	$R_{1,1}$	…	$R_{1,j}$	…	$R_{1,t}$
…	…	…	…	…	…
U_a	$R_{a,1}$	…	$R_{a,j}$	…	$R_{a,t}$
…	…	…	…	…	…
U_s	$R_{s,1}$	…	$R_{s,j}$	…	$R_{s,t}$

The matrix has s lines, which representatives of s user participation score, and the t columns represent the users score t items. $R_{i,j}$ represent score of user i scores items j, score range 1-5 points, reflecting the users of the items preference. If useri did not score the itemt j, then $R_{i,j} = 0$.

Recommended system denotes the prediction of matrix $\{PR_{i,ij},...PR_{i,ik}\}$ score estimated for the user U_i recommended, according user-item scoring matrix R by the similarity between users U={ U_1 , U_2 , ...,U_s }, and a collection of items M={ M_1 , M_2 , ..., M_s }. The estimated ranking $\{PR_{i,ij},...PR_{i,ik}\}$ is computed to predict { M_{ij} , ..., M $_{ik}$ },which is not scored, by the similarity of item-based, user-based and review-based scores.

3.1   User-Based Similarity Prediction

Definition 2. Similarity Measure includes the user-based, the item-based similarity, and the review-based similarity.

(1) the user-based similarity and the item-based similarity, we have modified the cosine formula[2] of similarity measure, for example, user similarity calculation is showed as following,
the similarity $Sim'(U_a, U_b)$ between the user U_a and U_b is:

$$Sim(U_a, U_b) = \frac{\sum_{M_i \in M'}(R_{a,i} - \bar{R}_a) \times (R_{b,i} - \bar{R}_b)}{\sqrt{\sum_{M_i \in M'}(R_{a,i} - \bar{R}_a)^2} \times \sqrt{\sum_{M_i \in M'}(R_{b,i} - \bar{R}_b)^2}} \tag{1}$$

$$Sim'(U_a, U_b) = \frac{min(|M'|, \gamma)}{\gamma} \times Sim(U_a, U_b) \tag{2}$$

Where M′ is the scoring item intersection ($M_{Ua} \bigcap M_{Ub}$) of user Ua and Ub, $\bar{R}_a, \bar{R}_b$ is the average score of user Ua and Ub, γ is the threshold number.

(2) the review-based similarity, review attributes for items provide additional clues about the underlying reasons for which a user review interested in particular items. We use the standard cosine formula to calculate similarity measure according to content of the reviews, the opinion and emotional score vectors can be substituted into the following formula:

$$Sim(C_a, C_b) = \frac{\sum_{i=1}^{k}(C_{a,i} \times C_{b,i})}{\sqrt{\sum_{i=1}^{k}(C_{a,i})^2} \times \sqrt{\sum_{i=1}^{k}(C_{b,i})^2}} \tag{3}$$

We introduce two concepts in uncertainty neighbor collaborative recommendation algorithms [2] as follow.

Definition 3 (Neighborhood Group). Neighborhood Group is most similar objects sets to the target collection. To the item goals and the user goals, the object in Neighborhoods group must meet similarity greater than the threshold value.

$$S(U_a) = \{U_x | Sim'(U_a, U_x) > \mu, a \neq x\} \tag{4}$$

$$S(M_j) = \{M_y | Sim'(M_j, M_y) > v, j \neq y\} \tag{5}$$

where, follow function values is the size of the two Neighborhood groups $S(Ua)$, $S(Mj)$: $|S(U_a)| = m |S(M_j)| = n$

Definition 4 (Trust Subgroup). Trust subgroup is the sub group with more restriction of Neighborhood group, not only to consider the similarity is greater than threshold, but also the value of common items is greater than a certain ε threshold by two users, and value of users score is greater than a certain δ threshold rated by two items.

$$S'(U_a) = \{U_x | Sim'(U_a, U_x) > \mu \& |M_{U_a} \cap M_{U_x}| > \varepsilon, a \neq x\} \tag{6}$$

$$S'(M_j) = \{M_y | Sim'(M_j, M_y) > v \& |U_{M_j} \cap U_{M_y}| > \delta, j \neq y\} \tag{7}$$

The size of two trust subgroups were recorded as: $|S'(U_a)| = m' |S'(M_j)| = n'$.

User-Based Recommendation. We study a class of user-based recommendation algorithms for users predictions. With neighborhoods group $S(U_a) = \{U_1, \ldots, U_m\}$ dynamically ensured for the target user's $U_x (1 \leq x \leq m)$, item's scoring can predict the project's score of target users. We get the predict user-based score $UB\_R_{a,j}$ based on user similarity. The formula is:

$$UB\_R_{a,j} = \bar{R}_a + \frac{\sum_{U_x \in S(U_a)} Sim'(U_a, U_x) \times (R_{x,j} - \bar{R}_x)}{\sum_{U_x \in S(U_a)} Sim'(U_a, U_x)} \tag{8}$$

Where the $\bar{R}_a, \bar{R}_x$ are scoring average of user U_a, U_x.

3.2 Item-Based Similarity Prediction

In this section we study a class of item-based recommendation algorithms for producing predictions to users. Unlike the user-based collaborative algorithm discussed the item-based approach looks into the set of items the target user has rated and computes how similar they are to the target item j and then selects n most similar items. At the same time their corresponding similarities are also computed. Once the most similar items are found, the prediction is then computed by taking a weighted average of the target user's ratings on these similar items. Item-based similarity prediction is dynamically determined the target item's $M_y (1 \leq y \leq n)$ by neighborhoods group $S(M_j) = \{M_1, \ldots, M_n\}$, user's scoring can predict the item's score of target users. We get the predict score$IB\_R_{a,j}$ based on user similarity. The formula is:

$$IB\_R_{a,j} = \bar{R}_j + \frac{\sum_{M_y \epsilon S(M_j)} Sim'(M_j, M_y) \times (R_{a,y} - \bar{R}_y)}{\sum_{M_y \epsilon S(M_j)} Sim'(M_j, M_y)} \tag{9}$$

Where the $\bar{R}_j, \bar{R}_y$ are scoring average of user M_j, M_y.

These methods computed the prediction on an item *i* for a user *u* by computing the sum ratings given by user on the items similarity. Each ratings is weighted by the corresponding similarity between items i and j. Although the collaborative recommendation algorithm has been widely used, there are still some shortcomings: user score data sparse and the cold start problem. The traditional methods can not effectively measure the similarity, which makes the calculated nearest neighbor is not accurate, when the user score data extremely sparse. The quality of the recommended algorithm decline overall, as user-based or item-based collaborative recommendation need to require the similarity measure between items and users. Meanwhile, the lack of a common score items, so many users can not be compared. So we proposed a multi-dimensional adaptive recommendation algorithm by extending opinion analysis to improve top-k recommendation.

4   Adaptive Multi-dimensional Recommendation

The adaptive multi-dimensional recommendation can solve the data sparseness problem by combination of user-based, item-based and review-based three recommended models, which it can improve the quality of top-k recommendation. First algorithm calculate the predicted score three dimensions of the dynamic metrics based on user similarity, item similarity and review similarity, and then automatically determines three weights value. Final it gets the goal prediction score of the target user.
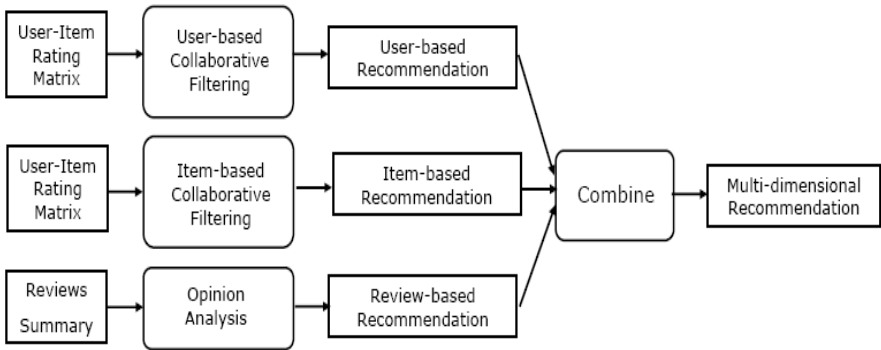
Fig. 1. Recommendation System Model

The system model can be divided into six parts: (1) get the first dimension prediction score based on user similarity; (2) get the second dimension prediction score based on item similarity; (3) get the third dimension prediction score based on review similarity; (4) to determine dynamic weight value of the three dimension data

distribution.; (5) to calculate final prediction score of the target user on the target item; (6) to do recommend according to the all prediction score ranking for the target users. System model is shown in Figure 1.

4.1 Extend Opinion Analysis

In many situations, opinion is directly related to sentiment. Product or movie reviews generally express whether a person liked or disliked the product or movie. Therefore, review analysis could be used to determine the document's opinion on a product or a movie. With general documents, however, opinion classification is capable of determining user opinion. For convenience presentation, we recommend specific example to introduce the film and carried out experiments. In the movie recommendation system, the film set M in every movie has a corresponding user reviews. The database is from the Internet Movie Database IMDB (Internet Movie Database, http://www.imdb.com), which user reviews of different content series is crawled down, table 1 shows the data size of the film commentaries:

Table 1. The scale of movie reviews

# total sentences	415454
# average sentences per review	247
# average tokens per sentence	19.6

Most research adopts a three-step approach to determining opinion. First, the feature words, opinion wordsand emotional tendency are classified, then the review score of the entire document based on an aggregate defined over the words is calculated and finally review score is taken to mean opinion. A sentiment polarity of a word is interpreted as the feeling people associate with the word.

In this paper, we examine the ability of a simple review-based analysis technique to mine different opinions. We consider three opinion classes, positive, neutral and negative opinions. We define completeness as an information retrieval system's ability to present the different opinions that exist for a given query. Finally, we use SentiWordNet [11] to determine the user-emotional score vector and judge emotions tend: the views of the word emotional polarity, which it will put <feature category, polarity> word into tuples.

For a movie review sentences, the above analysis steps can get all tuples of the <feature category, polarity>, we need to use these tuples to calculate the emotional score film on the feature at 6 category, the initial points of emotion = 0 for each category, scores is cumulative based on the following formula:

$$Score_k = \begin{cases} Score_k + 1 & if \quad polarity_k = "positive" \\ Score_k + 0 & if \quad polarity_k = "neutral" \\ Score_k - 1 & if \quad polarity_k = "negative" \end{cases} \tag{10}$$

Feature class represents the six feature word set {film, acting, storyline, cinematography, soundtrack, production } is defined as F ={F_1, F_2, …F_6}, so we can get the $t \times 6$ order rate matrix of a emotional characteristic scores, it's shown in Table 2 as below:

Table 2. Item-emotional score matrix

	F_1	…	F_j	…	F_6
M_1	$Score_{1,1}$	…	$Score_{1,j}$	…	$Score_{1,6}$
…	…	…		…	…
M_i	$Score_{i,1}$	…	$Score_{i,j}$	…	$Score_{i,6}$
M_j	$Score_{j,1}$	…	$Score_{j,j}$	…	$Score_{j,6}$
…	…	…		…	…
M_t	$Score_{t,1}$	…	$Score_{t,j}$	…	$Score_{t,6}$

where, score representatives item-emotional score matrix of the film obtained in the feature class emotional score; shaded in Table 2, respectively, two representatives of the emotional content of the film score and comments vector can be obtained similarity.

$$RB\_R_{a,j} = \frac{\sum_{Mz \in MUa} Sim(M_j, M_z) \times R_{a,z}}{\sum_{Mz \in MUa} Sim(M_j, M_z)} \tag{11}$$

4.2   Adaptive Recommendation Algorithm

Based on the above discussion, considering the three dimensions of information of target users Ua process item Mj provided by the rate projections, we get goal score prediction formula to be defined as:

$$PR_{a,j} = \alpha \times UB\_R_{a,j} + \beta \times IB\_R_{a,j} + (1 - \alpha - \beta) \times RB\_R_{a,j} \tag{12}$$

The algorithm intention is that you can still get the predicted score by using the opinion-based similarity, even if the user-based similarity and item similarity is not high. In practice recommendation system, due to the sparsity of the data rate, it's very easy happen to lead not high of user-based and item-based similarity. But opinion-based similarity score usually is high, because the users can easy access from the content-rich network, so review-based predicted score can compensate the lack of data sparseness. It is the main intent of multi-dimensional adaptive collaborative filtering algorithms, it can been seen from the formula (12).

Top-K Recommendation Algorithm:

```
Input : user-item scoring matrix R(s×t),
      user-review scoring matrix N(t×6),
      the threshold number γ,μ,ν,ε,δ, and
       controlled parameter ,θ
Output :  Top-K Recommendation Item list for object user U_a.
```

```
Calculated average user-based score: Uaverage;
Calculated average item-based score: Iaverage;
According  to  the  formula  (4)  get  user  U_a  of  neighbors
group
S(U_a )={U_1,…,U_m};
Calculated user U_a scored item sets M_(U_a )={M_1,…,M_q};
forj from 1 to t do
if U_a for item M_j have no score then
{ for i from 1 to m do
     if  U_i when item M_j scored then
     { down1 += userSim(U_a,U_i);
      p1 += userSim(U_a,U_i)*( R_(i,j)-Uaverage[U_i ]); }
     end if
  end for
 if down1==0 then UB_R_(a,j) = Uaverage[U_a ];
 else UB_R_(a,j) = Uaverage[U_a ]+up1/down1; end if
 According  to  the  formula  (5)  get  item  M_j  of  neighbors
              group S(M_j )={M_1,…,M_n};
for i from 1 to n do
     if U_a when item M_i scored then
        down2 += itemSim(M_j,M_i);
        up2 += itemSim(M_j,M_i)*( R_(a,i)-Iaverage[M_i ]);
     end if
end for
if down2==0 then IB_R_(a,j) = Iaverage[M_j ];
  else IB_R_(a,j) = Iaverage[M_j ]+up2/down2; end if
for i from 1 to q do
   down3 += reviewSim(M_j,M_i);
   up3 += reviewSim(M_j,M_i)*R_(a,i);
end for
if down3==0 then RB_R_(a,j) = 0;
       else RB_R_(a,j) = up3/down3;  end if
 calcWeight(a,b,c);
PR_(a,j)=a*UB_R_(a,j)+b*IB_R_(a,j)+c*RB_R_(a,j);
store(P,PR_(a,j));
} end if
end for
sort(P);  //decreasing order
return Top(k);
```

5   Experiment and Analysis

This section through the experiment, we test recommended quality of multi-dimensional adaptive collaborative filtering algorithms (MACF). UNCF is uncertain

neighborhood collaborative filtering recommendation algorithms. This experiment uses the MovieLens site (http://movielens.umn.edu) to provide the test data set, the site is a Web-based research-based recommendation system for receiving the user of the film's score and provide a list of recommended movies. Each user rated score of 1 to 5 points, rating at least 20 films, 5 points were very good, 1 means poor, different users have different film score, reflecting their interests and preferences.

User-film score rating of sparse matrix 1-100000 (943/1682) = 0.9370, indicating that this data set is quite sparse for the score matrix. We randomly assigned 80% of the entire data set used for training set and the remaining 20% for the test set. Dataset is randomly divided into5 times, so the formation of five different sets of training set and test set, these five data sets were recorded as dataset1, dataset2, dataset3, dataset4, dataset5. The size of datasets (1~5) is as below: users number=943, movies Number=1682, Training set=80000, Test set=20000.

Statistical accuracy metrics evaluate the accuracy of a system by comparing the numerical recommendation scores against the actual user ratings for the user-item pairs in the test dataset. Mean Absolute Error (MAE) between ratings and predictions is a widely used metric. MAE is a measure of the deviation of recommendations from their true user-specified values. For each ratings-prediction pair $<p_i, r_i>$ this metric treats the absolute error between them i.e., $|p_i-r_i|$ equally. The MAE is computed by first summing these absolute errors of the N corresponding ratings-prediction pairs and then computing the average. Suppose the user score prediction set is $\{p_1, p_2, ..., p_N\}$, corresponding to the actual rate set is $\{r_1, r_2, ..., r_N\}$. Formally, $\text{MAE} = \dfrac{\sum_{i=1}^{N} |p_i - r_i|}{N}$. The lower the MAE, the more accurately the recommendation engine predicts user ratings. Root Mean Squared Error (RMSE), and/or relation are also used as statistical accuracy metric.

Experimental parameters is dynamic: Through experiment, the purpose is to examine how to select the appropriate parameters for collaborative balance based on user, item and review on the recommendation results. dataset1 as the data set, first to make the same value, and adjustments to reconcile the size parameters, so that they increase from 10 to 100, the MAE values observed experimental results, shown in Figure 2.

Figure 2 horizontal axis shows the value of parameters, and the vertical axis indicates the value of the MAE results. It can be seen from the figure, when the change from 10 to 80, MAE continue from the increase to lower, and in the range 10 to 30 decreasing faster. When more than 80 hours, MAE and increasing with the increase, but the growth rate is very slow. Figure 2 shows that, if the parameter values are the same, the result was optimal. In order to verify the dynamic adaptive to determine the weight distribution method of performance, we will use the parameters to determine the weight method and the average weight method to do a comparison test, using dataset1 ~ dataset5 this five data sets, the results shown in Figure 3. Through this experiment, the aim is to test quality of multi-dimensional adaptive collaborative recommendation algorithms, verify that the algorithm takes into account multiple dimensions of predictive information after the data is able to alleviate the sparsely problem and improve the forecast system accuracy rate.
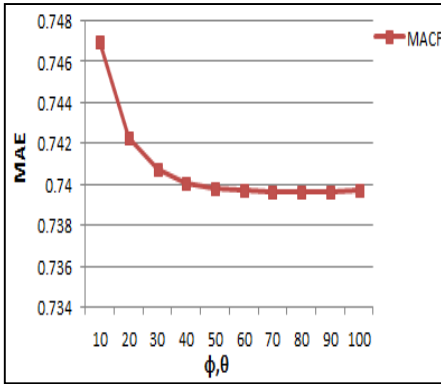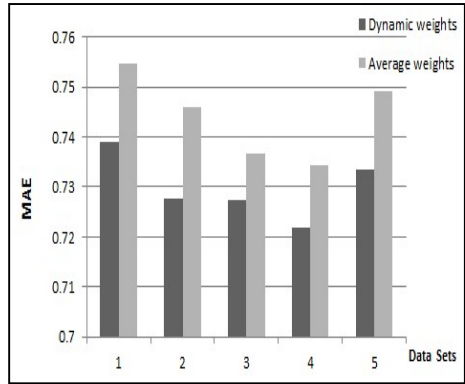
Fig. 2. Dynamic parameters

Fig. 3. The MAE contrast

The following is reasonable weights analysis of the user-based similarity and item-based similarity behind the decision, where:

$$\alpha = \frac{\phi \times M}{\phi \times M \ + \theta \times N \ + q}, \quad \beta = \frac{\theta \times N}{\phi \times M \ + \theta \times N \ + q},$$

In order to determine the optimal values of the case, we need one parameter fixed, so that the other parameters change, observed changes in MAE, here still use dataset1 as a data set.
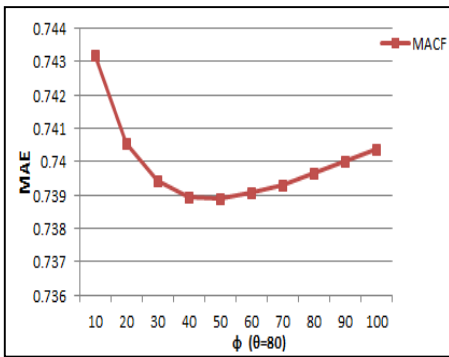
Fig. 4. Impact of ϕ on MAE when $\theta=80$
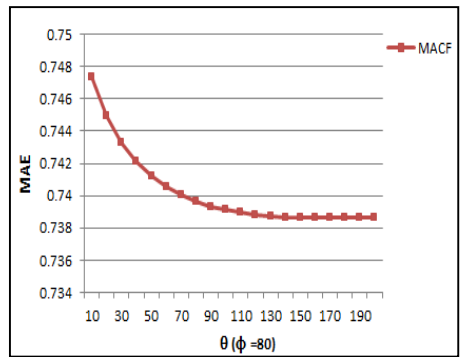
Fig. 5. Impact of θ on MAE when $\phi=80$

First of all the results are shown in Figure 4, they fixed increasing from 10 to 100. You can see, when between 10 to 50 changes, MAE decreases with the increase, and when greater than 50, MAE with the increase and continue to grow. Best seen when the experimental results. Then fixed, increasing from 10 to 200, the results shown in Figure 5. You can see, when between 10 to 160 changes, MAE decreases with the increase, and when the time is greater than 160, MAE with increasing increments. In fact, the optimal parameter values to reconcile with the specific data sets related to different data gathering produce different results of the optimal adjustment parameters, we can not make some of conclusions, but the data sets can be training

and calculations to obtain the optimal value of this parameter.Figure 4 parameters fixed, to reconcile the results of parameter changes fixed parameters in Figure 5, to reconcile the results of parameter changes.
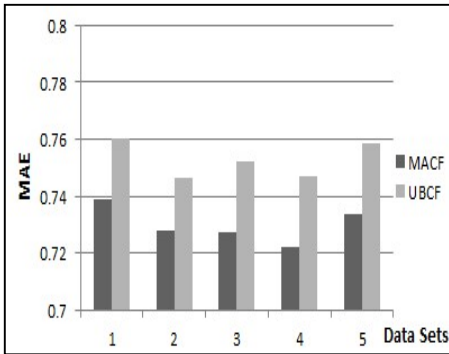
Fig. 6. Contrast of MACF & UBCF
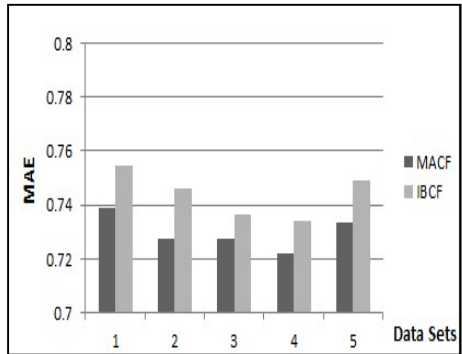
Fig. 7. Contrast of MACF & IBCF

The algorithm we only consider a single dimension with user-based collaborative filtering recommendation algorithm (UBCF), project-based collaborative filtering recommendation algorithm (IBCF) and comment on the collaborative filtering based recommendation algorithm (RBCF) were compared with recent years, the last to the experimental results on the performance of the uncertainty on better neighborhood collaborative filtering recommendation algorithms (UNCF) [2] were compared, each of comparative experiments are used dataset1 ~ dataset5 these five data sets, experimental results shown in Figure 6 to Figure 9.
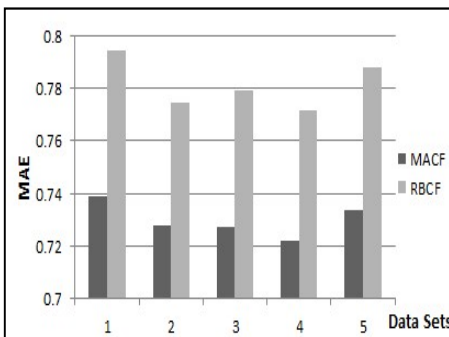
Fig. 8. Contrast of MACF & RBCF
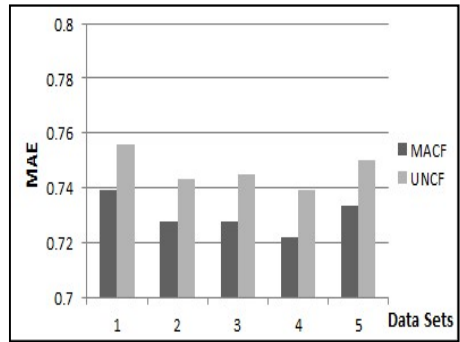
Fig. 9. Contrast of MACF & UNCF

6   Conclusions

In this paper, we proposed a multi-dimensional adaptive collaborative filtering recommendation algorithm by advanced traditional algorithm. The advanced collaborative filtering recommendation algorithm is combination of three recommended

model algorithm of user-based, item-based, and review-based similarity calculation based on comments, opinion mining technology. It can use dynamic measurement method to automatically determine three-dimensional weight of user-based, item-based, and review-based similarity, according to the strict definition, the target final prediction score is given by the target user. Experimental results show that multi-dimensional adaptive recommendation algorithm can effectively alleviate the dataset sparsely problem and achieve better prediction accuracy compared to other traditional collaborative recommendation algorithms.

References

1. Zhang, G.W., Li, D.Y., Li, P., Kang, J.C., Chen, G.S.: A collaborative filtering recommendation algorithm based on cloud model. Journal of Software 18(10), 2403–2411 (2007)
2. Huang, C.G., Yin, J., Wang, J., Liu, Y.B., Wang, J.H.: Uncertain neighbors' collaborative filtering recommendation algorithm. Chinese Journal of Computers 33(8), 1369–1377 (2010)
3. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Application of dimensionality reduction in recommender system-A case study. In: Proc. of the ACM WebKDD 2000 Workshop (2000), http://roboticsStanford.edu/-ronnyk/WEBKDD2000/
4. Zhao, L., Hu, N.J., Zhang, S.Z.: Algorithm design for personalization recommendation systems. Journal of Computer Research and Development 39(8), 986–991 (2002)
5. Aggarwal, C.C.: On the effects of dimensionality reduction on high dimensional similarity search. In: Proc. of the ACM PODS Conf. ACM, Santa Barbara (2001)
6. Deng, A.L., Zhu, Y.Y., Shi, B.L.: A collaborative filtering recommendation algorithm based on item rating prediction. Journal of Software 14(9), 1621–1628 (2003)
7. Papagelis, M., Plexousakis, D., Kutsuras, T.: Alleviating the Sparsity Problem of Collaborative Filtering Using Trust Inferences. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 224–239. Springer, Heidelberg (2005)
8. Jakob, N., Weber, S.H., Müller, M.-C., Gurevych, I.: Beyond the Stars: exploiting free-text user reviews to improve the accuracy of movie recommendations. In: TSA 2009, Hong Kong, China, November 6 (2009)
9. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004), pp. 168–177 (2004)
10. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of IJCAI, Hyderabad, India, pp. 1606–1611 (January 2007)
11. Esuli, A., Sebastiani, F.: SENTI-WORDNET: A publicly available lexical resource for opinion mining. In: Proceedings of LREC (2006)
12. Papagelis, M., Plexousakis, D.: Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. Engineering Applications of Artificial Intelligence 18, 781–789 (2005)
13. Liu, B.: Sentiment analysis and subjectivity. In: Handbook of Natural Language Processing, 2nd edn. (2010)

*C*DDTA-JOIN: One-Pass OLAP Algorithm for Column-Oriented Databases

Min Jiao[1,3], Yansong Zhang[2], Yan Sun[1,3], Shan Wang[1,3], and Xuan Zhou[1,3]

[1] DEKE Lab, Renmin University of China, Beijing 100872, China
[2] National Survey Research Center at Renmin University of China, Beijing 100872, China
[3] School of Information, Renmin University of China, Beijing 100872, China
{zhangys_ruc,swang}@ruc.edu.cn

Abstract. Row-store commonly uses a volcano-style "once-a-tuple" pipeline processor for processing efficiency but looses the I/O efficiency when only a small part of columns are accessed in a wide table. The academic column-store usually uses "once-a-column" style processing for I/O and cache efficiency but it has to suffer multi-pass column scan for complex query. This paper focuses on how to achieve the maximal gains from storage models for both pipeline processing efficiency and column processing efficiency. Based on the "address-value" mapping for surrogate key in dimension table, we can map incremental primary keys as offset addresses, so the foreign keys in fact table can be utilized as native join index for dimensional tuples. We use predicate vector as bitmap vector filters for dimensions to enable star-join as pipeline operator and pre-generate hash aggregators for aggregat based on the column. Using these approaches, star-join and pre-grouping can be completed in one-pass scan on dimensional attributes in fact table, and the following aggregate column scanning responses for the sparse accessing aggregation. We can gain both I/O efficiency for vector processing and CPU efficiency for pipeline aggregating. We perform the experiments for both simulated algorithm based on the column and the commercial column-store database.

Keywords: OLAP, *C*DDTA-JOIN, predicate-vector, column-store.

1 Introduction

The databases are divided into two types, OLTP and OLAP. OLTP uses tuple as update and transaction granularity with row-oriented storage model, OLAP commonly specifies a small part of columns for aggregating with column-oriented storage model. Row-oriented model uses volcano-style "once-a-tuple" pipeline processing while column-oriented model uses "once-a-column" style processing such as MonetDB BAT algebra[1]. Pipeline processing is based on the traditional cost model to minimize the temporary data cost. Because of the low selectivity predicates, the early materialized row-oriented pipeline processing only produce small result sets. If a certain node in query plan tree involves much latency, the whole pipeline processing efficiency will be affected greatly. Furthermore, pipeline processing increases local

dataset in pipeline streaming, which causes higher cache miss for the large data size. Column-oriented model divided relational algebra into fined granularity by atomic column operations. The column algebra makes processing dataset even smaller for better cache performance. Without pipeline processing, column-oriented model uses OID as intermediate join index for join correlated columns. In the complex queries, column joins produce large intermediate cost and processing cost.

In the view of storage model and processing model, the SQL engine can be divided into four types, row-oriented storage model with row processing model(type 1), row-oriented storage model with column processing model(type 2), column-oriented storage model with column processing model(type 3) and column-oriented storage model with row processing model(type 4). Type 1 and type 3 can be considered as "native" query processing engine, and type 2 and type 4 can be regarded as "enabled" query processing engine. In real applications, on-the-fly storage model conversion is widely adopted by database vendors such as Greenplum, Vertica, InfoBright, etc. to achieve better I/O efficiency and get the compatible processing engine.But the combined processing model(pipeline processing+column processing) is still absent because the early materialization(row-oriented processing) and late materialization(column-oriented processing) are hard to merge together. This paper focuses on how to exploit the advantages for both pipeline processing and vectorized processing, our contributions are as two-folds:

1. We exploit the performance bottleneck of multi-pass scan for column processing.
2. We propose pipeline processing with column-oriented model with late materialization for both processing efficiency and I/O efficiency.

The related work is presented in Section 2. Column-oriented model based *C*DDTA-OLAP model is discussed in Section 3. Section 4 shows the results of experiments. Finally, Section 5 summarizes this paper.

2   Related Work

2.1   Storage Model

Row-oriented storage model organizes physical attributes of tuple consecutively. The page-slot structure needs many pointers to locate tuples having variable length in the block. The pointer can locate the tuple efficiently with additional space cost. The projection operation on the row tuple which only need a small part of attributes causes high overhead, and this operation is commonly used in analytical processing.

Column-oriented storage model makes same attribute values being stored together, the various length attribute stores values in a large continuous space with offset pointers for each various length item and the attribute column only store the fixed length offset pointers. The fixed length values enable column-store model can locate values by offsets without pointers which is widely used in the page-slot structure, thus the storage efficiency is improved. Column-oriented model organizes same type values

together which can apply compression techniques to reduce space cost[2] proposed a compression technique with row-oriented model by C-Table. C-Table is a tablet style compression table with standard relational supports, which can transform inner compression into relational operations.

PAX[3] is a hybrid storage model inside the block i.e. *n* tuples inside the block are stored as columns. When update the tuples only produce one I/O cost and only necessary columns are accessed in the buffered blocks, it's a tradeoff between update cost and memory bandwidth. InfoBright[4] divides table into row groups with 64K tuples, in each row group the tuples are stored as column-oriented model. This strategy can be viewed as horizontal partition on column-oriented database. In each row group, columns have the same offsets, so the columns can be converted into row tuples according to positions in buffer blocks on-the-fly. Upon the column-oriented model, row tuple processing engine can be utilized to keep compatibility with traditional SQL engine. This strategy only optimizes the I/O performance by column store and compression, but the processing optimization is left to the traditional SQL engine.

Data Morphing[5] organizes correlated attributes as combined columns for better cache locality. The key issue is how to exploit the correlations among attributes, if the correlations are changed for other workloads, the re-organizing overhead is much higher. For popularly used star schema in OLAP, star-join between fact table and dimension tables is the basic operator, the correlation of foreign keys in fact table is high. Vertica[6] uses "projection" as division unit, projections are designed by workloads. In general view, projections should be defined on-the-fly according to diverse workloads.

2.2   Processing Model

OLAP queries are multidimensional queries, a typical OLAP query always groups and aggregates result tuples by joining the tuple in the fact table and correlated dimensional tuples together. The OLAP query can be summarized as SPJGA operation(S: select, P: project, J: join, G: grouping, A: aggregate). For SP operation, column store model can get high performance for its I/O or memory bandwidth efficiency, the results of single column are identified with OID pairs[1] or bitmap[7] for further operation. For JGA operation, row-wise model with pipeline processor seems to have the advantage of the column-store. In different scenarios, column-oriented model and row-oriented model have diverse features.

For traditional row-oriented databases, project operation is invoked on buffered blocks and is a costly operation for picking specified items from the physical row-wise layout. PAX supports efficient in-memory physical projections with the inner column-oriented model, but the I/O cost for big data of PAX equals to that of traditional row store. Column-oriented model improves projection efficiency for both I/O and memory bandwidth for only physically accessing specified columns, upon the storage model, there are two major roadmaps for query processing engine. The first roadmap is traditional row-wise processing engine with on-the-fly DSM(Decomposed Storage Model) and NSM(N-array Storage Model) conversion such as Greenplum, ParAccel, InfoBright, C-store[8]. DSM means the column store model and NSM

means the row-wise model. Column blocks are converted into row-wise tuples in the buffer and pushed to the traditional row-wise SQL engine for query plan tree to process. The performance majorly depends on I/O efficiency. The second roadmap uses column-oriented query processing engine such as MonetDB, SybaseIQ[9], invisible-join[7]. Column-oriented query processing produces much intermediate data for column correlations, and the query plan is much more complicated than traditional query plan based on pipeline. Row-wise engine is simple but the pipeline processing will lead to high miss ratio and column-wise engine benefits from high cache performance but it needs complex column operations.It's a dilemma to choose performance or complexity. [10] proposed the pre-processing technique for blocks with on-the-fly DSM/NSM conversion, which is based on *memcpy,* helps to make storage model adapt to processing model. The main consideration is whether the cache performance gains exceed the overhead of dynamic conversion.

Another considering factor for choosing storage model is hardware feature. When the working datasets are memory resident, on-the-fly DSM/NSM conversion is low overhead. When the datasets are disk resident, on-the-fly DSM/NSM conversion means striped scan on disk which may involve high overhead because of disk head conflicting. If the hardware is SSD or RAID, the random accessing and parallel accessing features enable the DSM/NSM conversion on-the-fly to be efficient. So storage model is influenced by multiple factors, such as processing engine, performance, and hardware features, all those are all involved to make an overall consideration.

3   DSM to NSM-OLAP Model

3.1   NSM Model

NSM equals to the column-store model. Figure 1 illustrates the pipeline processing for row-wise OLAP. The tuples in the fact table are iteratively piped through hash tables from dimensions, only tuples that are satisfied all the predicates will win the game for final aggregating. The early materialization guarantees the one-pass scan on tables and also makes tuples longer when they are piped through the hash filters. In the low selectivity queries, most of the data bandwidth is useless.
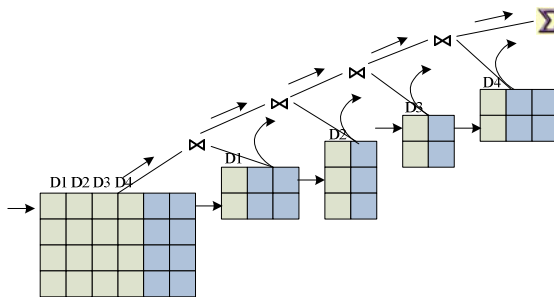
Fig. 1. Pipeline processing for row-wise OLAP

Pipeline processing benefits from no intermediate data cost and lose efficiency for valid data stream. The length of pipeline 2/3(amount of hash filters) and filter over-head(hash probing cost) dominate the total efficiency of pipeline processing. We can further divide the hash filter as join filter and candidate grouping datasets to make filters even smaller and use late materialization[7] to reduce bandwidth consumption for useless data.

3.2   DSM Model

DSM model performs once-a-column style processing, and the column joins are basic operations for correlated columns. The selectivity of OLAP is usually much higher than OLTP, the selectivity on certain dimension may exceeds 80%, so column joins between fact table and dimensions may produce high overhead. We perform further analysis on MonetDB and invisible-join of C-store to exploit the performance bottle-necks for DSM model.

Figure 2 illustrates the processing phases of invisible-join[7] with a typical star-join between fact table and three dimensions. In first phase, predicates are applied on dimensions for creating hash filters. In second phase, each foreign key column joins with correlated hash filters to generate bitmap vector that is used to identify the join results for each dimension, the bitmaps are performed with bit operation of AND(&) for global result vector. In the last phase, the global bitmap vector merges with each foreign key column to extract final matched foreign keys for probing dimensional attributes.
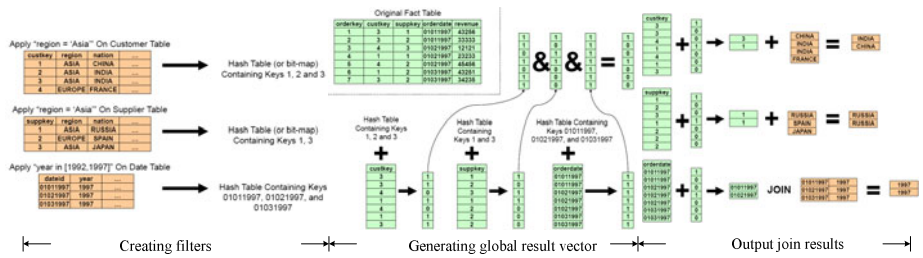
Fig. 2. Analysis for invisible-join

Compared with classical MonetDB OID based column algebra, invisible-join use bitmap instead of OID BAT joins for better performance. In our view, there are four performance issues in the algorithm:

1. Predicate selectivity may be high on dimension table, so the hash table may be large with high hash probing overhead.
2. The join result bitmaps are produced on the fact table's columns which are large in size with big dataset, each bitmap operation is $O(n)$ cost.
3. The foreign key columns in fact table are scanned twice for partial and global join results, the memory bandwidth consumption or I/O cost is high. The measure columns in the fact table also involve $O(n)$ cost bitmap filtering.

4. [7] uses positional join on the grouping attributes in the probing phase, and this is on the observation that the primary keys of dimension table can be naturally mapped to the offset addresses of dimensional columns. The hash filter constricts the simple but efficient positional join mechanism to be applied in the star-join phase for further optimization.

The lessons we learn from invisible-join are that we should inherit pipeline processing from row-wise processing to eliminate multi-pass scan on columns and we should make use of positional join for star-join phase to simplify multi-stage joins of column processing model.

3.3   DDTA-JOIN Model

DDTA-JOIN (Directly Dimensional Tuple Accessing Join)[11] is our research work for main-memory OLAP. The algorithm is based on the rule of surrogate key in data warehouse. The dimensions are commonly increasing slowly and usually utilize the surrogate key (natural increment keys such as 1,2,3…) as primary key. The surrogate key can be directly mapped to the offset address of dimension columns, so the foreign keys in the fact table not only represent join keys but also represent the addresses of memory resident dimension columns. We also proposed predicate-vector as bitmap filters for star-join. Figure 3 illustrates the process of the same star-join as figure 2. Predicates are applied on dimension tables for predicate-vectors which use bitmap to identify the positions of filtered dimensional tuples. Compared with invisible-join, we use bitmap on small dimension tables not on big fact table, the space cost for bitmaps is much lower. Predicate-vectors take the place of hash filters for filtering join tuples, the bitmap filter size is smaller than hash filters. Furthermore, positional join on bitmap filters can be proved to be constant processing time in spite of diverse selectivity while the performance of hash probing varies very much with different selectivity. We use row-oriented model for fact table and column-oriented model for dimension tables, the star-join is performed as pipeline processor by positional join on predicate-vectors, the grouping attributes are late materialized by positional join in the last filtering phase to avoid useless data accessing in row-wise pipeline processing.

We use the SSB (Star Schema Benchmark, the test benchmark improved on the TPC-H) as our application scenarios. The non-surrogate key of dateid in date dimension is incremental sequence with *month-day-year* format, we can on-the-fly map date key into surrogate key by calculating the interval days from current date to the beginning date.
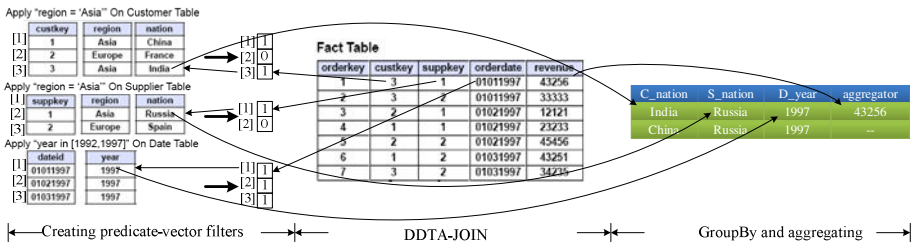

Fig. 3. Example for DDTA-JOIN

DDTA-JOIN is a simple processing model with hybrid storage model, we have gained remarkable performance improvements for in memory OLAP and multi-core OLAP. In this paper, we continue our research on uniform column-oriented model for better memory bandwidth and I/O efficiency.

3.4   *C*DDTA-JOIN Model

*C*DDTA-JOIN represents DDTA-JOIN with column-oriented model(*C*-Column). For memory resident fact table, columns can be on-the-fly converted as row-wise tuples by positions, we perform the pipeline processing with predicate-vector filters and late materialization mechanism for GroupBy operation just like figure 3.
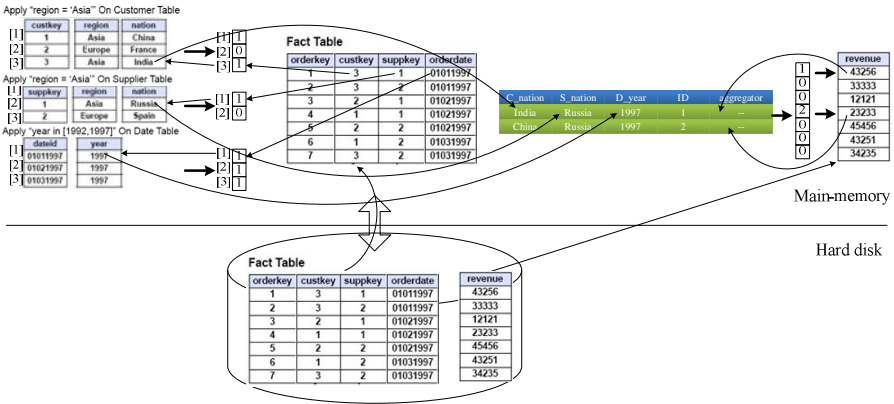
Fig. 4. Example for *C*DDTA-JOIN

For disk-resident fact table, we have to tradeoff the I/O efficiency of column scan and the additional intermediate data cost for column joins. We focus on two mechanisms for disk resident column-oriented fact table:

1.   Hybrid column store for fact table
We regard foreign key attributes as high correlated attributes for OLAP based on the star-join, the foreign key attributes are stored with Data Morphing style as columnar. During scanning on the columnar, the star-join based on the predicate-vector and the grouping operations based on the late materialization are performed to generate the hash aggregate slots with empty aggregate attributes, each aggregate slot is assigned with a slot ID. A result vector like invisible-join is created to identify the filtered status for each fact tuple, if the fact tuple goes through all the filters for final output, it either locates an existed hash slot or create a new hash slot for future aggregate and marks the vector position with hash slot ID as figure 4 shows. After scanning the columnar, the measure columns can directly use result vector for aggregating, the measure value with correlated non-zero vector value is pushed into the specified hash aggregate slot with same hash slot ID in vector for aggregating.

2.   Column store for fact table
Hybrid storage model is too heavy for databases to support two storage engines with different architecture and compression mechanisms. Column store databases use

on-the-fly DSM/NSM conversion for the project operation, we consider the following scenarios:

(1) Naïve columnar. Foreign key columns are dynamically accessed as project operation to generate result vector, then the aggregating task on the measure columns by the column scanning is finished. If the aggregate operation involves multiple measure attribute like SUM ($M1$-$M2...$), measure columns $M1$, $M2$,… are accessed in same project operation for aggregating, the I/O efficiency relies on the column storage engine and hardware feature.

(2) Combined surrogate key. We can further compress multiple foreign keys into single combined surrogate key, for example, there are four foreign keys ($DF1$,$DF2$,$DF3$,$DF4$) in fact table, the maximal binary length of four keys are $n1$,$n2$,$n3$,$n4$, we can encoding the four keys into single encoding key with $n1+n2+n3+n4$ bits, the first $n1$ bits represent $DF1$, the next $n2$ bits represent $DF2$, and so on. The encoding key is added to fact table for new column as combined surrogate key, and we can directly scan the combined surrogate key column for star-join and pre-aggregate instead of project operation with multiple foreign key columns.

Column store makes the costly column sum operation and column update operation much cheaper than row store, if the updates on dimensions break the rule of surrogate key, we can re-organize primary key in dimension table and update correlated foreign key column in fact table to guarantee the key-address mapping. The combined surrogate key column can be considered as an embedded join index for star-join, the join index size and update cost are all cheaper than traditional index.

Through these approaches, we implement one-pass scan OLAP with CDDTA-JOIN algorithm, and the foreign key columns scanning needs efficient star-join and pre-aggregate, measure columns scanning needs efficient random scan for aggregate with low selectivity.

4   Experiments

Our experiments are conducted on a Lenovo R630 G7 server with four Intel® Xeon® CPU E7420@2.13GHz CPUs, 32GB DDR II SDRAM, and Dawning RAID DS8340FF 16×16000 rpm SATA, having total storage volume of 5TB. The OS is AXS3 X86_64, Linux version 2.6.18-128.7AXS3. We produce dataset with SF=32 for both memory resident and disk resident CDDTA-JOIN testing by standard SSB data generator. CDDTA-JOIN only optimizes join with GroupBy clause, so we neglect Q1.x queries, which are without GroupBy clauses in SSB. CDDTA-JOIN is developed with C++ language, and we also implement the invisible-join algorithm with same coding style to make two algorithms comparable.

In order to give a comprehensive analysis, we design three experimental scenarios to verify performance gains of CDDTA-JOIN.

4.1   *C*DDTA-JOIN in Memory Resident OLAP

We design three storage models of memory resident fact tables for CDDTA-JOIN experiments.

(1) NSM model. We divide fact table in SSB into vertical partitions, which means 9 active attributes are organized into one table, we neglect other inactive attributes in performance experiment.
(2) DSM model. Each fact table attribute is stored as an array, we on-the-fly convert all the arrays into rows by array indexes. Each attribute in one logical tuple has different memory addresses.
(3) Hybrid model. We organize four foreign keys attributes in fact table as row table, and leave measure attributes as arrays.

CDDTA-JOIN is linear and one-pass scan OLAP algorithm, the performance is dominated by table scan performance. In memory resident scenario, table scan efficiency lies on cache line efficiency, if one cache line can carry more data for processing, the cache miss may be reduced. Cache line efficiency of NSM is about 2/3(about 6 attributes are involved in 9-attribute row), DSM model suffers from about 6 cache line misses for processing one row, the potential cache miss may be higher, hybrid model increases cache line efficiency in foreign key columnar but loses cache line efficiency in measure column accessing. Figure 5 shows the experimental results of each algorithm.
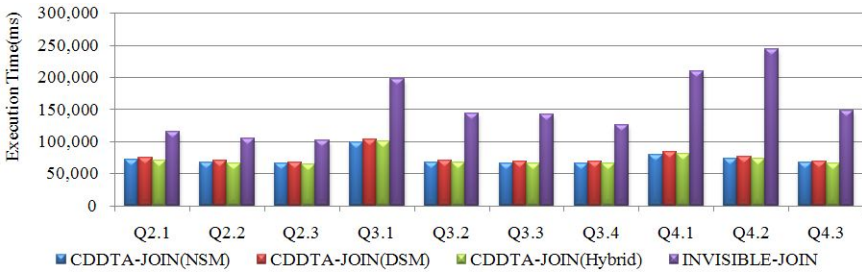
Fig. 5. CDDTA-JOIN in memory resident OLAP

Invisible-join costs more time than other algorithms for its twice scan on fact columns. Hybrid model CDDTA-JOIN wins the test, NSM model costs 0.31% more time than hybrid mode and DSM model costs 5.09% more time than hybrid model, the performance difference is dominated by memory address layout of memory resident fact table.

As a result, CDDTA-JOIN is prior to invisible-join in column processing model for its efficiency in one-pass scan OLAP.

4.2 *CDDTA-JOIN in Disk Resident OLAP*

We use open-source PostgreSQL as storage engine, and stores each measure attribute as single table to simulate column store. Hybrid model is designed with one vertical partition table with four foreign key attributes and several single tables as measure columns. The storage efficiency is not as good as pure columns store, because we need the low layer API to generate result vector on which random scan on measure

columns is based. For the two candidate algorithms, the storage efficiency is equal. Figure 6 shows the total results, the reasons for performance gap between *C*DDTA-JOIN and invisible-join include fact bitmap cost, bitmap operation cost, re-scan on foreign columns cost, etc. *C*DDTA-JOIN in Q4.x takes longer time than other groups, one reason is that Q4.x involves 6 fact columns for processing while Q2.x and Q3.x only involve 4 fact columns, and the other reason is, aggregate operation in Q4.x is based on algebra operation of two measure columns, we have to concurrently read two fact columns with I/O synchronizing cost.
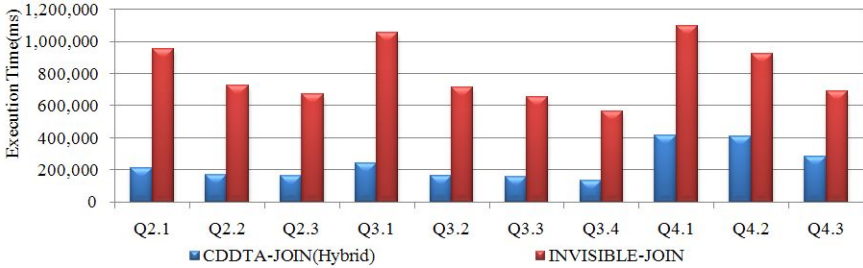
Fig. 6. *C*DDTA-JOIN in disk resident OLAP

4.3   *C*DDTA-JOIN on Column Store Database

Hybrid store *C*DDTA-JOIN is suitable for OLAP but has difficulties with main stream column store databases. We design experiment for TPC-H Q5 with SF=10 on an open-source column-oriented database to test the performance gains in real column oriented database for *C*DDTA-JOIN. TPC-H Q5 is a typical star-join query with 3 dimensions, we set memory resident dimension tables as public hash tables for concurrent queries, while the dimension loading cost can be regarded as single-usage cost for query workloads. Column store applies a column-oriented storage model for storage engine with compression technique to provide perfect I/O efficiency, upon the storage engine is on-the-fly DSM/NSM conversion based on row-wise processing engine. In TPC-H Q5 query cost analysis, join cost occupies about 43% in total cost. Sequence scan means we perform the *C*DDTA-JOIN relying on low layer API for project operation, foreign key columns in fact table are sequentially scanned for star-join processing, and measure columns are also sequentially scanned with result vector for final aggregation. However, if the total selectivity is low, most of measure column I/O is useless. Positional scan employs the low layer API to support random scan on column by position, so we can use result vector as an index to access non-zero position value on measure columns, as a result, the I/O efficiency for measure column is improved. Figure 7 illustrates the running time for three algorithms, we can see that on-the-fly DSM/NSM conversion based on column store only improves I/O efficiency in the data fetching process, while the join cost is still a big performance issue like traditional row store database. The loading dimensions cost can be dropped as system pre-processing cost, and we can further overlap processing cost and column I/O cost by asynchronous I/O to enable memory processing and I/O fetching as a pipeline.
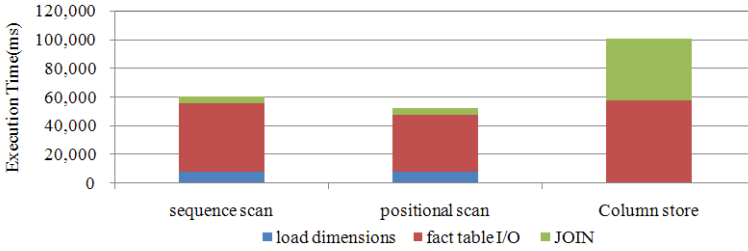
Fig. 7. CDDTA-JOIN for TPC-H Q5

5   Conclusions

Column-oriented storage model is well accepted as analytical database storage engine. The overall performance is dominated by two levels, one is the storage model, the other is the query processing model. In storage level, column-oriented model suits for low projection rate analytical workload, database can gain maximal I/O efficiency by only fetching the specified columns instead of fetching all the data like row-oriented model. In the query processing level, column-oriented model is double-sided sword for processing engine. On one side, column processing makes column join very efficient for its smaller size than traditional pipeline processing model, and the memory bandwidth and cache performance are all improved. On the other side, column operation has no context for the following column operations, much intermediate data are materialized for correlated column operations. Join operation is the most frequent operation among columns, and join cost still dominate the whole performance. For SPJGA analytical operation, GroupBy is another performance bottleneck for the twice scan on join columns like invisible-join, for disk resident database, the I/O cost is doubled. So row-wise processing and column-wise processing have their own pros and cons, neither of them are the final answer for high performance processing.

Our philosophy is "divide and conquer" rule. Traditional hash filter in pipeline includes both join filter and GroupBy values, and the selectivity of join filter is high for the dimensions while the GroupBy values are valid in a much lower global selectivity at the end of pipeline, so we divide the join filter and GroupBy values into two parts, so join filter is optimized as bitmap predicate-vector with efficient positional join, and GroupBy operation is pushed to the final stage as late materialization by the positional join on grouping dimensional columns. By these approaches, we can convert foreign key in fact table as row-wise tuple on-the-fly, and perform pipeline processing with column-oriented predicate-vector filtering, and fetch the GroupBy values from column-oriented dimension tables based on the final filtering results on-the-fly. For disk-resident database, we create result vector and pre-aggregate hash table, and the result vector carry both filtering information and aggregate slot information, so we can perform a vectorized hash aggregation for both I/O efficiency and CPU efficiency.

As a special statement, all these approaches are based on the feature of surrogate key, and the natural sequence enables join key as memory offset address. The surrogate key is commonly used in data warehouse and we can naturally take the advantage of

surrogate key for star-join optimization. For primary key in dimensions being not surrogate key such as *dateid* in date dimension of SSB, we can add the additional surrogate key column for both dimension and fact table, and the column alternation cost is much lower in column-oriented database because only a few columns are accessed, but it is unacceptable in row-oriented databases because column alternation usually leads to full table being re-organized. So our surrogate key based *C*DDTA-JOIN can be applied in general purpose data warehouse scenarios. Column-oriented model and positional join in memory make *C*DDTA-JOIN gain both I/O efficiency and CPU efficiency.

References

1. Boncz, P.A., Mangegold, S., Kersten, M.L.: Database architecture optimized for the new bottleneck: Memory access. In: VLDB, pp. 266–277 (1999)
2. Bruno, N.: Teaching an Old Elephant New Tricks. In: CIDR 2009, Asilomar, California, USA (2009)
3. Ailamaki, A., DeWitt, D.J., Hill, M.D.: Data page layouts for relational databases on deep memory hierarchies. The VLDB Journal 11(3), 198–215 (2002)
4. Ślęzak, D., Wróblewski, J., Eastwood, V., Synak, P.: Brighthouse: An Analytic Data Warehouse for Adhoc Queries. In: PVLDB 2008, August 23-28 (2008)
5. Hankins, R.A., Patel, J.M.: Data morphing: an adaptive, cache-conscious storage technique. In: Proceedings VLDB, pp. 417–428 (2003)
6. The Vertica Analytic Database: Rethinking Data Warehouse Architecture. WinterCorporation White Paper (May 2005)
7. Abadi, D.J., Madden, S.R., Hachem, N.: Column-Stores vs. Row-Stores: How Different Are They Really? In: Proceeding of SIGMOD 2008, Vancouvrer, BC, Canada (2008)
8. Stonebraker, M., Abadi, D.J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., O'Neil, E.J., O'Neil, P.E., Rasin, A., Tran, N., Zdonik, S.B.: C-Store: A Column-oriented DBMS. In: Proceedings of the VLDB, Trondheim, Norway, pp. 553–564 (2005)
9. MacNicol, R., French, B.: Sybase IQ Multiplex -Designed for analytics. In: Proceedings of VLDB (2004)
10. Zukowski, M., Nes, N., Boncz, P.A.: DSM vs. NSM: CPU performance tradeoffs in block-oriented query processing. In: DaMoN 2008, pp. 47–54 (2008)
11. MOSS-DB: A Hardware-Aware OLAP Database. In: WAIM 2010, pp. 582–594 (2010)
12. O'Neil, P., O'Neil, B., Chen, X.: The Star Schema Benchmark (SSB), http://www.cs.umb.edu/~poneil/StarSchemaB.PDF

A Study of the Single Point Mutation Loci in the Hepatitis B Virus Sequences via Optimal Risk and Preventive Sets with Weights

Qi Zhang[1,*], Junpeng Zhang[1,*], Jianmei Gao[2,*], Jianfeng He[1,**], Xinmin Yan[2,**], Lei Ma[1], Xianwen Zhang[1], and Jiuyong Li[3,**]

[1] Kunming University of Science and Technology, Kunming, China
[2] Institute of Basic Medicine of the First People's Hospital of Yunnan Province, and Center of Clinical Molecular Biology, and Kunhua Affiliated Hospital of Kunming Medical College, Kunming, China
[3] University of South Australia, South Australia, Australia
zhqq1122@126.com, zhangjunpeng508@gmail.com,
{gaojm7609,yxmin08,xianwen-zhang0717}@163.com,
jfenghe@kmust.edu.cn, roy_l_murray@hotmail.com,
jiuyong.li@unisa.edu.au

Abstract. HBV (Hepatitis B Virus) infection is a severe global health problem. In recent years, the single point mutation as an essential element in the HBV evolution has been extensively studied, however, only the limited mutation loci were reported. In this paper, we proposed a new method to apply MORE (Mining Optimal Risk PattErn sets) and RPSW (Risk and Preventive Sets with Weights) algorithms to study the single point mutation loci in the HBV sequences. Experimental results show that the proposed approach is efficient to mine mutation loci, such as the reported mutation loci at positions ntT1753C, ntA1762T, ntG1764A, nt1896, and the new found mutation loci at positions ntA1436G, ntG1629A, ntA1383C, ntA1573T, and the risky of positive mutation loci at positions nt1726, nt1657, nt1463, nt1658, nt1498, nt1386. Furthermore, the proposed method is also able to find out highly relevant association rules or patterns based on the feature mutation loci.

Keywords: Hepatitis B virus, Feature selection, Optimal risk and preventive patterns, Mutation loci.

1 Introduction

HBV belongs to a family called hepadnaviridae which is closely related with DNA virus. This kind of virus infection is a severe global health problem and a common cause of liver disease and cancer. It is estimated that at least 2 billion living people

---

have been infected with HBV, as many as 378 million of individuals have chronic infection and approximate 620,000 people die each year from acute and chronic sequelae of HBV infection [1, 2]. Moreover, 4.5 million new HBV infections occur worldwide each year, of which a quarter progress to liver disease [3].

The viral genome of HBV is a partially double-stranded circular DNA of approximate 3.2bp that encodes four overlapping open reading frames (ORFs: *S*, *C*, *P* and *X*). The *S* ORF encodes the hepatitis B surface antigen (HBsAg) and can be structurally and functionally divided into the *pre-S*1, *pre-S*2, and *S* regions, the *C* ORF encodes the hepatitis B e antigen and core protein, the *P* ORF encodes the polymerase protein, the *X* ORF encodes the *X* protein [4]. The HBV genome can be classified into 9 genotypes: A, B, C, D, E, F, G, H and I. Recently, HBV genotypes are defined based on at least 8% divergence across the complete genome sequence [5] and less than 4% intra-genotypic divergence [6]. The distribution of HBV genotypes varies in different geographical regions and accounts for differences in the prevalence of HBV variants in different parts of the world.

The causes for the large variations in the natural history of HBV infection are not fully understood, but some scientists revealed that the variations are associated with the virological itself factors, host immunological factors, genetic factors and experimental factors [7-9]. In recent years, the virological and immunological factors of HBV have been extensively studied, but the examination of the relationships between host genetics and HBV resistance is still in its infancy [10-12]. This study was focus on finding the mutation loci which contribute to the study of SNPs in HBV infection. Although the single point mutation as an essential element in HBV evolution has been studied, only the limited mutation loci were reported, such as the positions ntA1762T and ntG1764A [13]. At the present, single nucleotide polymorphism(SNP) discovery relies mostly on direct DNA sequencing or on denaturing high performance liquid chromatography(dHPLC). They have evolved from labor intensive, time consuming, and expensive processes to some of the most highly automated, efficient, and relatively inexpensive methods. In this paper, a new method was proposed to apply MORE and RPSW algorithms to study the single point mutation loci about the genetic factors in *pre-C* and *X* (nt1374~nt1900) regions of HBV sequences. Our results partly agreed with the previous report, for example, the mutation loci at positions ntT1753C, ntA1762T, ntG1764A and nt1896. Besides that, we also found the new mutation loci and the risky of positive mutation loci, such as the positions ntA1436G, ntG1629A, ntA1383C, ntA1573T, nt1726, nt1657, nt1463, nt1658, nt1498, nt1386.

The rest of the paper is organized as follows. We preprocessed the experiment data sources and made a review of related methods including information entropy-based feature selection, optimal risk and preventive patterns and optimal risk and preventive sets with weights in section 2. The experimental results were presented and analyzed in Section 3. Finally, the differential analysis of hepatitis B virus was concluded in Section 4.

2   Materials and Methods

2.1   Experiment Data Sources

In this study, the data set was gathered from HBV databases of the First People's Hospital of Yunnan Province. The data includes 10 serum samples of ten patients with four positive and six negative. HBV DNA was extracted from serum by phenol-chloroform extraction as the template of PCR. Samples were amplified with the primers by PCR to get the ~526bp HBV gene fragment, the data sources contain the ORFs: *pre-C* and *X* (nt1374~nt1900). After being purified, the PCR product was cloned into pUC18 vector and the recombinant plasmids were transformed into competent cell of JM109. The positive colony was identified by restriction endonuclease and sequencing in both ways. For each sample, 10 to 60 clones were sequenced directly and 364 HBV sequences were attained in all. An example of the data set is illustrated in Table 1.

We obtained the full experiment data set of HBV sequence data that contains 364 sequences, which include 623 base sites. In this paper, base loci correspond to attributes (623 attributes) and each sequence reflects to a sample. The class attribute contains two types: positive and negative. The positive class where replication is very active in the body with 155 sequences, the negative class indicates the human body having some immunity to the hepatitis B e antigen with 209 sequences. The mapping process is shown as Figure 1.

2.2   Method

In this paper, there are three steps to constrain and summarize HBV sequences. Firstly, we find out the feature loci which are most useful for classifying samples. Furthermore, optimal risk and preventive patterns about these feature loci are developed with MORE algorithm [14]. Finally, RPSW algorithm [15] is used to make a differential analysis of hepatitis B virus.

2.2.1   An Information Entropy-Based Feature Selection

An important step in data pre-processing is feature selection. It is necessary to restrict the number of input attributes and produces good predictions. The selection of relevant attributes and elimination of irrelevant ones are the key question in machine learning. One would like to use those attributes that are "relevant" to the target predictions.

The feature section method we use is an information gain based, and it selects the most informative attributes for classifying examples. Given a set of class attribute S, including both positive and negative samples. The entropy of S is defined as [16]:

$$Ent\left(S\right) = -p_{\oplus} \log_2 p_{\oplus} - p_{\odot} \log_2 p_{\odot}. \tag{1}$$

Table 1. The HBV data set, where records are classified as two distinct categories, positive and negative

Samples ID	HBV Sequences	Class
10001	ATGGCTG~…TTTGGGG~CATG	negative
10002	ATGGCTG~…TTTGGGG~CATG	negative
......	……	…….
8047	ATGGCTG~…TTTGGGG~CATG	positive
8048	ATGGCTG~…TTTGGGG~CATG	positive
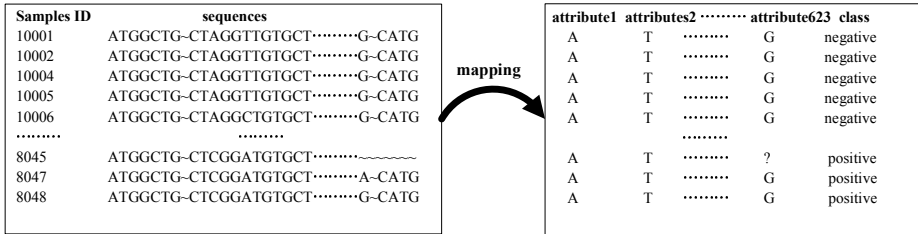
Fig. 1. Schematic diagram of the original data source mapping

Where, $p_\oplus$ and $p_\ominus$ are the proportions of positive and negative samples in S, respectively. The value of $\log_2 0$ is defined as 0 in all calculations. For example, if all the attributes of S belong to the same class, the entropy is 0. If the set contains an equal number of positive and negative samples, the entropy is 1. When the collection contains unequal numbers of positive and negative examples, the entropy is between 0 and 1.

Information gain is a measurement of the effectiveness of an attribute in classifying the training data, it is only the expected reduction in entropy caused by dividing the samples according to this attribute [16]. Gain(S, A) of an attribute A, relative to a set of samples S, is defined as:

$$Gain(S, A) = Ent(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Ent(S) \qquad (2)$$

Where, the *Values* (A) is the set of all possible values for attribute A, and S_v is the subset of S. Note that the first part is the entropy of the original set S which is mentioned in Equation (**1**), and the second part in Equation (**2**) is the entropy after S is partitioned using attribute A. The partition entropy is the sum of the entropies of each subset S_v, weighted by the fraction of samples $\frac{|S_v|}{|S|}$ that belong to S_v. Therefore, Gain(S, A) is the reduction in entropy caused by knowing the value of attribute A. The value of Gain(S, A) is the number of bits saved when the target value of an arbitrary member of S has being encoded, by knowing the value of attribute A.

This method has been implemented in Weka workbench [17].

2.2.2   Optimal Risk and Preventive Patterns

Patterns are defined as a set of attribute values pairs and it represented R. In the following, we define positive class as p, negative class as n. In the public HBV data, the proportion between positive and negative is larger than its real proportion. However, the proportion of positive should be much less than negative. Therefore, the set of *support* in HBV data is different from other data. We use a *lsupp* (local support) [14] to decide whether a pattern is frequent or not. The *lsupp* is described as follows:

$$lsupp(R \rightarrow p) = \frac{supp(R \wedge p)}{supp(p)} \qquad (3)$$

A pattern is called frequent if its *lsupp* is greater than a given threshold.

   RR (Relative Risk) or *OR* (Odds Ratio) is one of the metrics often used in epidemiological studies [18], which is a concept for the comparison of two groups with respect to a certain undesirable event. For example, if R is attribute value pair $328 = A$, the class is positive, and *OR*=3.0, then this means that when attribute value pair $328 = A$ are three times more likely to be positive than they are not. When *OR* =1, it indicates that the factor have no effect on the incidence of a disease. When *OR* is higher than 1, it shows that the factor is likely to be a risk factor, and *OR* is lower than 1, it illustrates that the factor is likely to be a preventive factor. The *RR* measurement is more conservative than the *OR*. If *RR* is higher than a given threshold, the pattern is more likely to be a risk pattern. Otherwise the pattern is more likely to be a preventive pattern. An example on how to calculate *RR* and *OR* is given as follow.

   As being illustrated in Table 2, outcomes are categorized as positive (+) or negative (-), HBV is labeled as having (+) or not (-) having a certain HBV under the investigation. Let a and c be the number of HBV in positive (+) and negative (-), b and d be the number of no-HBV in positive (+) and negative (-), respectively.

$$RR(HBV(+)) = \frac{a}{a+b} \div \frac{c}{c+d} = \frac{a(c+d)}{c(a+b)}$$

$$OR(HBV(+)) = \frac{a}{b} \div \frac{c}{d} = \frac{ad}{bc} \qquad (4)$$

Mining risk and preventive patterns can bring redundant patterns which are not helpful to observe the results. For example, we have two risk patterns, {"Caffeine =yes" and "Cancer=no"} with *RR* as 4.1, {"Caffeine=yes", "Smoking=no" and "Cancer=no"} with *RR* as 4.0. Actually, the latter pattern with lower *RR* than the former pattern when it incorporated factor "Smoking=no". It can be deduced that the former pattern is stronger than the latter pattern. Optimal risk and preventive patterns are set of all strong patterns. Optimal risk and preventive patterns can be mined by MORE (Mining Optimal Risk PattErn sets) algorithm to exclude superfluous patterns [14].

   Optimal risk and preventative patterns are extracted from risk and preventive patterns. On the one hand, optimal risk pattern set includes all risk patterns that have higher relative risk than of their sub-patterns. On the other hand, optimal preventive

patterns include all preventive patterns that have lower the relative risk that in case is less than a given threshold. When the sub-patterns of risk and preventive patterns cannot satisfy this requirement, these risk and preventive patterns are ignored.

Table 2. The pattern is composed of some factors for judging HBV or not

Factor	Positive (+)	Negative (-)	Total
HBV(+)	a	b	$a + b$
HBV(-)	c	d	$c + d$
Total	$a + c$	$b + c$	$a + b + c + d$

2.2.3 Differential Analysis Based on Optimal Risk and Preventive Sets with Weight

Based on optimal risk and preventive patterns, RPSW algorithm [15] calculates the frequency of each attribute value pair and sort all attribute value pairs by the frequency decreasing order. The attribute value pairs in optimal risk patterns and the attribute value pairs in preventive patterns may be considered as risk factors and preventive factors set, respectively. If an attribute value pair is a frequent element of optimal risk or preventive patterns and its frequency is greater than or equal to the given expected frequency threshold, then it belongs to risk or preventive set.

In optimal risk and preventive sets, only optimal risk patterns whose relative risk is higher than relative risk threshold in optimal risk patterns, and optimal preventive patterns whose relative risk is lower than relative risk threshold in optimal preventive patterns, are selected to generate optimal risk and preventive sets, and there is no common set between optimal risk and preventive sets.

For example, suppose we have five risk patterns (the relative risk threshold: 2.0):

Pattern 1: $\{R_1, R_2\}$ (RR=4)
Pattern 2: $\{R_1, R_2, R_3\}$ (RR=4.3)
Pattern 3: $\{R_2, R_5\}$ (RR=6)
Pattern 4: $\{R_2, R_4\}$ (RR=1.6)
Pattern 5: $\{R_3, R_4, R_5\}$ (RR=1.3)

These risk patterns involve five patterns and five attributes: R_1, R_2, R_3, R_4 and R_5. As discussed earlier, only the first three patterns are selected. The selected risk patterns involve four attributes: R_1, R_2, R_3 and R_5. If the attribute value pairs still exist in preventive patterns, we can compare the grade of membership of them in risk and preventive sets, and then determine whether these attribute value pairs belong to risk or preventive factors.

Optimal risk and preventive pattern sets with weights are based on optimal risk and preventive sets. We only consider attributes in optimal risk and preventive sets. To make results more intuitive, we normalized the weight of each attribute value pair. The total weights of optimal risk and preventative weight are 100 respectively. Each attribute value pair in optimal risk and preventive sets has a weight, which generates optimal risk and preventative factor sets with weights respectively.

3   Experiment Results

3.1   Optimal Risk and Preventive Patterns

One problem of HBV sequence data is the large number of attributes involved, it is necessary to remove those irrelevant attributes. The measure of feature selection we used is information gain, and it is simply the expected ranking in entropy caused by different thresholds and sorts all the information gain by calculating Equation (**2**). We set the thresholds as T=0.05, 0.1, 0.15, 0.2. The feature attributes were attained respectively as shown in Table 3:

Table 3. The relative risk threshold as T=0.05, 0.1, 0.15, 0.2. Feature attributes were attained respectively.

Threshold	#Attributes
0.05	59
0.10	36
0.15	19
0.20	12

3.2   Optimal Risk and Preventive Patterns of Mutation Locus

Hepatitis B virus data in this paper are described by 623 general attributes, 209 in class negative and 155 in class positive. Here we set the minimum local support as 0.43, select the relative risk threshold as 1.5 and 0.67 for optimal risk and preventive patterns, respectively, and the length of the attribute is L=7. It returned 420 patterns including 388 optimal risk patterns and 32 optimal representative patterns, respectively. The number of the most representative optimal risk and preventive patterns is 13.

The following are the first three optimal risk patterns with the highest relative risk.
Optimal Risk Pattern 1: $RR = 6.5000$

- attribute299 = A
- attribute132 = G
- attribute327 = G

Optimal Risk Pattern 2: $RR = 6.5000$

- attribute299 = A
- attribute132 = G
- attribute328 = A

Optimal Risk Pattern 3: $RR = 6.5000$

- attribute69 = G
- attribute132 = G

- attribute618 = G
- attribute327 = G

The following are the first three optimal preventive patterns with the lowest relative risk.

Optimal Preventive Pattern 1: $RR = 0.0000$

- attribute209 = A
- attribute11 = A
- attribute93 = G

Optimal Preventive Pattern 2: $RR = 0.0000$

- attribute209 = A
- attribute11 = A
- attribute124 = C

Optimal Preventive Pattern 3: $RR = 0.0000$

- attribute209 = A
- attribute11 = A
- attribute302 = A

To obtain the most valuable patterns, we present the most representative patterns from the optimal risk and preventive pattern sets. For example, Optimal Risk Pattern 1 for positive in optimal risk patterns including three attributes "attribute299 = A" "attribute132 = G" and "attribute327 = G", its relative risk is 6.5000. Optimal Preventive Pattern 1 in optimal preventive patterns including one attribute "attribute209 = A" "attribute11 = A" and "attribute93 = G", its relative risk is 0.

3.3   Differential Analysis of Hepatitis B Virus Sequence

The weight of each attribute is from the percentage of each attribute in optimal risk and preventive set. We assume the total weight is 100, and the weight of each attribute is the percentage of each attribute multiplying 100. The weight of each attribute is used to indicate how important each attribute is, and using it, we can know a patient's risk and preventive weight.

Figure 2 lists the weight of each locus in risk and preventive set of HBV. We can see the largest weight of locus is "attribute305=T" (13.1278) in optimal risk set, while the largest weight of locus is "attribute209=A" (28.8462) in optimal preventive set. As described in section II, if attribute305=T, the HBV sequence is at risk of positive and if attribute209=A, the HBV sequence is at preventive of negative. Furthermore, the common loci present in Figure 2 (a) and (b), we can divine from the loci which may be the mutations. For example, attribute69=G in risk set and attribute69=A in preventive set, it means that this position at nt1436 occurred the amino-acid substitution ($G \rightarrow A$).
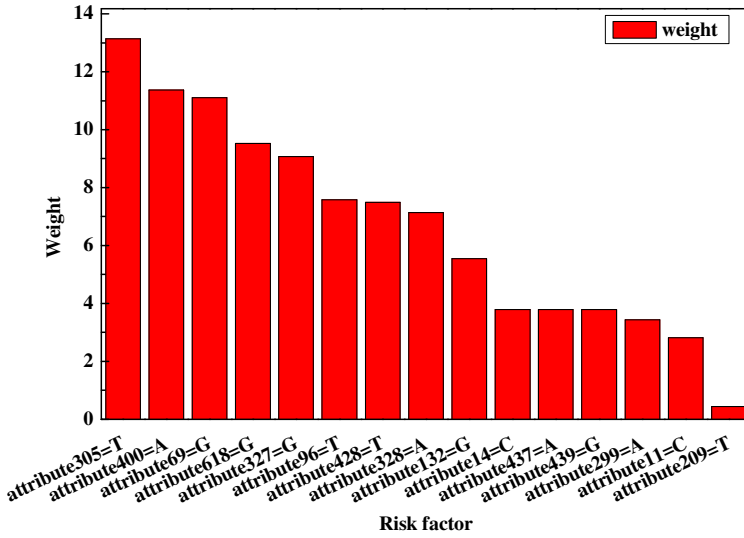
(a) The weight of attribute value pairs in optimal risk set
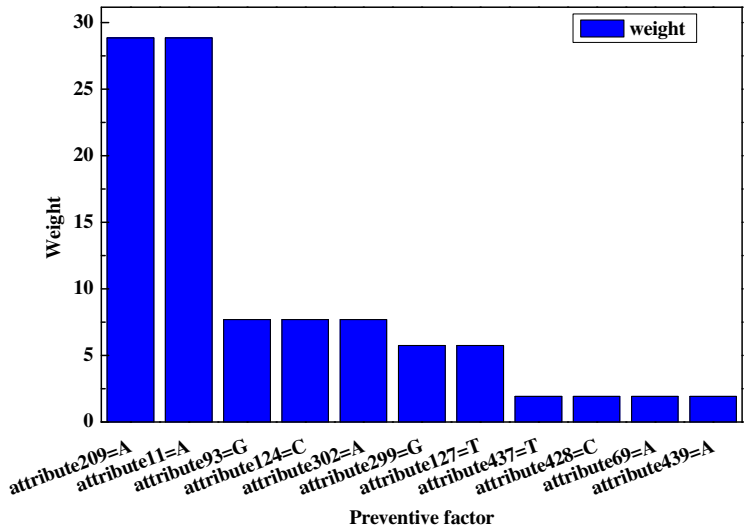
(b) The weight of attribute value pairs in optimal preventive set

Fig. 2. The weight of attribute value pairs in optimal risk and preventive set, where *lsupp*=0.43, *T*=0.15 and *L*=7

Here the minimum local support as 0.43, the relative risk threshold as 1.5 and 0.67 for optimal risk and preventive patterns, respectively. Under the conditions, most found loci are of great interest to domain experts and verified by them as shown in Table 4, such as the mutation loci at positions ntT1753C, ntA1762T, ntG1764A and

nt1896 have been reported in the precise studies [13, 19, 20]. Moreover we also discovered some new mutation loci, and the risky of positive mutation loci which in HBV sequences are vulnerable to mutation. The new mutation loci at positions ntA1436G, ntG1629A, ntA1383C, ntA1573T, and the risky of positive mutation loci at positions nt1726, nt1657, nt1463, nt1658, nt1498, nt1386. They will provide useful reference for medical practitioners.

Table 4. The relative risk threshold as 0.15 for optimal risk patterns, 0.67 for optimal preventive patterns. The position corresponding to feature attributes in HBV sequences. The weight is the frequency of each attribute value pair and sort all attribute value pairs by the frequency decreasing order which in optimal risk and preventive patterns.

Threshold=0.15					
Attribute value pairs in risk patterns			**Attribute value pairs in preventive patterns**		
Risk factor	**Position**	**Weight**	**Preventive factor**	**position**	**Weight**
attribute305=T	nt1635	13.1278	**attribute209=A**	**nt1573**	**28.8462**
attribute400=A	nt1726	11.3656	**attribute11=A**	**nt1383**	**28.8461**
attribute69=G	**nt1436**	**11.1013**	attribute93=G	nt1460	7.6923
attribute618=G	nt1896	9.5154	attribute124=C	nt1490	7.6923
attribute327=G	nt1657	9.0748	attribute302=A	nt1632	7.6923
attribute96=T	nt1463	7.5771	**attribute299=G**	**nt1629**	**5.7692**
attribute428=T	**nt1753**	**7.4890**	attribute127=T	nt1493	5.7692
attribute328=A	nt1658	7.1366	**attribute437=T**	**nt1762**	**1.9231**
attribute132=G	nt1498	5.5507	**attribute428=C**	**nt1753**	**1.9231**
attribute14=C	nt1386	3.7885	**attribute69=A**	**nt1436**	**1.9231**
attribute437=A	**nt1762**	**3.7885**	**attribute439=A**	**nt1764**	**1.9231**
attribute439=G	**nt1764**	**3.7885**			
attribute299=A	**nt1629**	**3.4361**			
attribute11=C	**nt1383**	**2.8194**			
attribute209=T	**nt1573**	**0.4405**			

4   Conclusions

The number of single point mutation discovery methods has exploded in recent years and many robust methods are currently available. But the new approaches must be developed to lower the cost and increase the speed of detection. In this paper, we constrained and summarized HBV sequences data with optimal risk and preventive sets with weights. According to experimental results, the most important finding in

our study is that we not only discover some reported mutation loci, but also reveal a number of interesting rules or patterns based on the mutation loci in *pre-C* and *X* regions. Besides the published mutations in the precise studies, we also found new mutation loci and risky of positive mutation loci in HBV sequences. For example, the new mutation loci at positions ntA1436G, ntG1629A, ntA1383C, ntA1573T, and the risky of positive mutation loci at positions nt1726, nt1657, nt1463, nt1658, nt1498, nt1386. Whether these mutation loci are really meaningful to diagnose disease or not, it needs further clinical research. Experimental results show that the method is useful for exploratory study on large HBV sequence data sets. It quickly finds some "mutation spots" in the HBV sequence data set and can be used to generate and refine hypotheses for further time consuming statistical studies. The results will gain insight into the study of hepatitis B virus.

References

1. William, M., Lee, M.D.: Hepatitis B Virus Infection. New England Journal of Medicine 337(24), 1733–1745 (1997)
2. Goldstein, S.T., Zou, F., Hadler, S.C., Bell, B.P., Mast, E.E., Margolis, H.S.: A mathematical model to estimate global hepatitis B disease burden and vaccination impact. International Journal of Epidemiology 34, 1329–1339 (2005)
3. Zanetti, A.R., Van Damme, P., Shouval, D.: The global impact of vaccination against hepatitis B: A historical overview. Vaccine 26(49), 6266–6273 (2008)
4. Ganem, D.E., Schneider, R.J.: Hepadnaviridae: the viruses and their replication. In: Blaine Hollinger, F. (ed.) Viral Hepatitis (2001)
5. Norder, H., Couroucé, A.M., Coursaget, P., Echevarria, J.M., Lee, S.D., Mushahwar, I.K., Robertson, B.H., Locarnini, S., Magnius, L.O.: Genetic Diversity of Hepatitis B Virus Strains Derived Worldwide: Genotypes, Subgenotypes, and HBsAg Subtypes. Intervirology 47, 289–309 (2004)
6. Kramvis, A., Kem, M.C.: Relationship of genotypes of hepatitis B virus to mutations, disease progression and response to antiviral therapy. Journal of Viral Hepatitis 12(5), 456–464 (2005)
7. Luo, K.X.: Hepatitis B: Basic biology and clinical science, pp. 56–70. People's Medical Publishing House (2001)
8. Ferrari, C.: Hepatitis B Virus Immunopathogenesis. Annual Review of Immunology 13(1), 29–60 (1995)
9. Abel, L., Dessein, A.J.: The impact of host genetics on susceptibility to human infectious diseases. Current Opinion in Immunology 9(4), 509–516 (1997)

10. Dean, M., Carrington, M., O'Brien, S.J.: Balanced polymorphism selected by genetic versus infectious human disease. Annual Review Genomics Hum. Genetics 3, 263–292 (2002)
11. Adrian, V., Hill, S.: The immunogenetics of human infectious diseases. Annual Review Immunology 16, 593–617 (1998)
12. Hill, A.V.: Host genetics of infectious diseases: old and new approaches converge. Emerging Infectious Disease 4(4), 695–697 (1998)
13. Takahashi, K., Aoyama, K., Ohno, N., Iwata, K., Akahane, Y., Baba, K., Yoshizawa, H., Mishiro, S.: The precore/core promoter mutant ($T^{1762}A^{1764}$) of hepatitis b virus: clinical significance and an easy method for detection. Reading, ROYAUME-UNI: Society for General Microbiology 76, 3159–3164 (1995)
14. Li, J., Fu, A.W.-C., Fahey, P.: Efficient discovery of risk patterns in medical data. Artificial Intelligence in Medicine 45, 77–89 (2009)
15. Zhang, J., He, J., Li, J., Ma, L.: Constraining and Summarizing Optimal Risk and Preventive Patterns in Medical Data. In: 4th International Conference on Bioinformatics and Biomedical Engineering, pp. 1–4. IEEE Press, Chengdu (2010)
16. Mitchell, T.M.: Machine learning. McGraw-Hill Science/Engineering/Math. (1997)
17. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. SIGKDD Explorations 11(1), 10–18 (2009)
18. Gange, S.J., Cole, S.R.: Epidemiologic Analysis - A Case-oriented Approach. Oxford University Press, New York (2002)
19. Jake Liang, T., Hasegawa, K., Rimon, N., Wands, J.R., Ben-Porath, E.: A Hepatitis B Virus Mutant Associated with an Epidemic of Fulminant Hepatitis. The New England Journal of Medicine 324(24), 1705–1709 (1991)
20. Okamoto, H., Tsuda, F., Akahane, Y., Sugai, Y., Yoshiba, M., Moriyama, K., Tanaka, T., Miyakawa, Y., Mayumi, M.: Hepatitis b virus with mutations in the core promoter for an e antigen-negative phenotype in carriers with antibody to e antigen. Journal of Virology 68(12), 8102–8110 (1994)

Enforcing Interaction and Cooperation in Content-Based Web3.0 Applications

Antonio Bevacqua[1], Marco Carnuccio[1], Alfredo Cuzzocrea[1,2],
Riccardo Ortale[2], and Ettore Ritacco[2]

[1] University of Calabria, Cosenza, Italy
{abevacqua,mcarnuccio}@deis.unical.it, cuzzocrea@si.deis.unical.it
[2] ICAR-CNR, Cosenza, Italy
{ortale,ritacco}@icar.cnr.it

Abstract. In this paper we complement our previous contribution [2] where we introduced principles and functional architecture of *Borè*, a novel framework that truly realizes Web3.0 principles and tools for the next-generation Internet. In particular, in this paper, we provide a more comprehensive overview of *Borè* by detailing peculiar aspects, such as the query languages and the related operators that allow us to browse and query the graph-like structure underlying a typical *Borè*'s view, and providing a complete case study on *Borè* in action that focuses on an application domain related to the University context.

Keywords: Web Architecture, Semantic Web, Social Network, User Profiling, Content Management System.

1   Introduction

Web-browsing models aim to pursue a challenging goal, i.e., to turn the Web environment into an intelligent and proactive setting, in which users play a key role. This is known as the *Semantic Web* and involves the development of advanced Web interfaces, capable to enrich with semantics both the supplied information and the users' activities, with the ultimate goal of offering customized access and support to each individual user.

Nowadays, the not yet definitive Web3.0 model, which extends the Web1.0 and Web2.0 initiatives (e.g., [11]), tries to further evolve the Web into the personal *universe* of each user, thus introducing the concept of *portable personal Web*, that follows from the widespread adoption of new technologies and devices. The idea is to generate adaptive systems, such as [4, 6], which record and analyze users' activities, in order to define suitable user profiles, whose knowledge is in turn useful to anticipate their preferences, expectations and tastes. As a matter of fact, the single individual becomes the core of the Web, thus making the *folksonomy* evolve in the *me-onomy*.

Based on these motivations, in this paper we complement our previous contribution [2] where we introduced principles and functional architecture of *Borè*,

a novel framework that truly realizes Web3.0 principles and tools for the next-generation Internet. In particular, in this paper, we provide a more comprehensive overview of *Borè* by detailing peculiar aspects, such as the query languages and the related operators that allow us to browse and query the graph-like structure underlying a typical *Borè*'s view, and providing a complete case study on *Borè* in action that focuses on an application domain related to the University context.

2   Overview of the *Borè* Framework

The intuition behind *Borè* is to exploit some principles of the object-oriented programming [12,10] in the context of Web-application development. The logic model or prototype of a Web application is viewed as a directed graph $\mathcal{G}$, whose nodes are the resources, while edges denotes the relations among pairs of such resources. By resource we mean all those entities that pertain to the Web application: a Web page, media contents, users, communities and so forth. A link between two resources, e.g., two Web pages, is a relation modeled as a direct edge from the former resource to the latter one.

By following the ideas of semantic Web systems, that exploit definition languages such as OWL[1], RDFS[2] and RDF[3], Web information can be generalized by structuring raw data in high-level content objects.

More precisely, each resource (resp. relation) is a pair $\{type, instance\}$: *type* is the general definition, i.e. the schema, of the resource (resp. relation) and it is composed by a set of *field*s, whereas *instance* is the type instantiation. A field is an atomic information, whose base type can be any among integer, real, string and so forth. The single inheritance property [12,10] holds for both resources and relations. A resource (resp. relation) that inherits from another resource (resp. relation) acquires all of information within the latter in addition to its own. The separation in types and instances and the inheritance property bring the significant advantages of object-oriented software development in the design and implementation of Web applications. We here mention module reuse, simplified code development, ease of extension and maintenance and compactness of shared information.

Formally, the prototype $\mathcal{G}$ of a Web application in *Borè* is a tuple $\mathcal{G} = \{R, E, RTT, ETT, RT, ET\}$, whose constituents are introduced next. R is the set of all resources corresponding to the nodes of $\mathcal{G}$. E is the set of all relations, i.e., the set of edges of $\mathcal{G}$. RTT and ETT are, respectively, the resource and the relation type taxonomies: resource and relation types are stored in two dedicated taxonomies enabling type inheritance. The root of RTT is the type *Object*, while the root of ETT is *Edge*. RT and ET are two forests of taxonomies. Each taxonomy in RT (resp. ET) is an inheritance hierarchy among resource (resp. relation) instances. RTT, ETT, RT and ET are useful constituents, that allow

---

[1] http://www.w3.org/TR/owl2-overview/
[2] http://www.w3.org/TR/rdf-schema/
[3] http://www.w3.org/RDF/

a simple management of Web browsing in terms of navigation operators as it will be described in sec. 4.

In our formalization, the interaction of a user with a Web application can be seen as a visit on the prototype $\mathcal{G}$: by clicking on a link within the current Web page (which corresponds to following a relation in the graph model of the Web application, that departs from the resource being currently experienced), the user moves to another Web page (i.e. visits another resource of the graph model).

The strengths following from the adoption of the foresaid logic model for Web applications deployed within *Borè* are manifold. First, a simplified resource management and querying. In particular, query processing in response to user interactions can take advantage of the solid results and foundations in the field of graph theory [3]. Second, the possibility to dynamically define new resources and types. Third, different Web applications can share the same schema (the set of all types) and differ only in their respective instances. Fourth, compactness and cooperation. As a matter of fact, different Web applications may share some resources, thus avoiding redundancies. Moreover, resource sharing allows the generation of social cooperations and social networks among users. Fifth, it is easy to store and analyze users' click streams, in order to understand their behavior and tastes.

To exemplify the foregoing concepts, we introduce in fig. 1 the graph (or logic model) of a toy Web application. The resources pertaining to the Web application (i.e., an institute, two workers and a student) are represented by nodes. Edges between nodes correspond to relations between resources. Actually, the graph in fig. 1 depicts an institute, with two workers and a student, that can publish news and organize events.
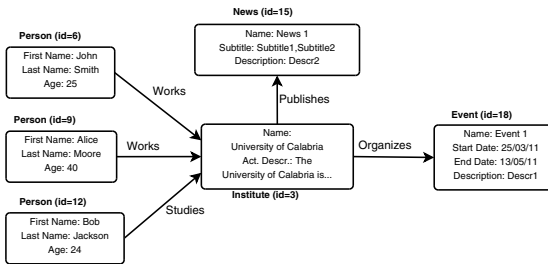
Fig. 1. The graph model of a toy Web application

3   *Borè* Architectural Details

The architecture of the *Borè* platform, shown in fig. 2, is designed around the Model-View-Controller (MVC) design pattern [5]. This allows to separate the

three essential aspects of an application, i.e. its business, presentation and control logic with the purpose of considerably reducing both time and costs for development and maintenance. The architectural components of the *Borè* platform included within the Model, View and Controller layers are indicated in fig. 2. More specific details on the modular MVC architecture are provided below.
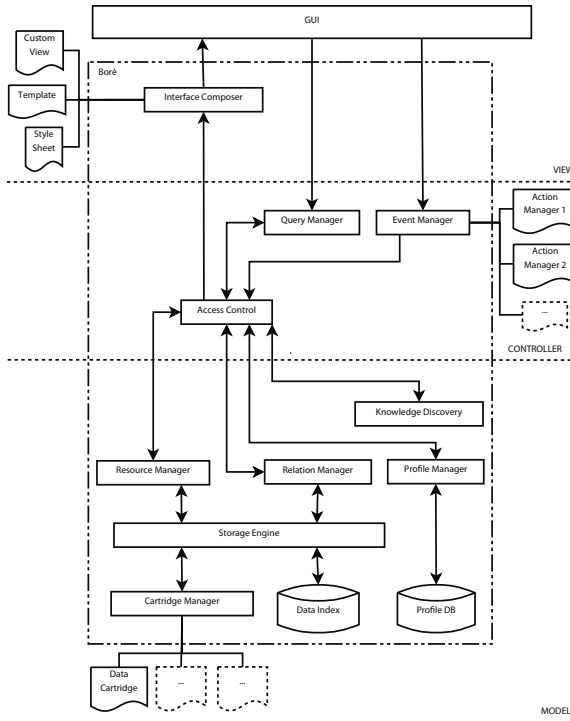

Fig. 2. The anatomy of the *Borè* platform

View. The *View* is responsible both for the navigation of the graph of a Web application and for query-result visualization. However, being in a Web environment, the final rendering of the information delivered by any Web application deployed within *Borè* is delegated to the user *browser*, which is the actual *GUI*. Instead, query answers are delegated to the *Interface Composer* module, that can communicate with the rest of the architecture.

The Interface Composer relies on some pluggable, customizable and extensible modules, that can be categorized into the following three types: (i) *Template* is the module type, that specifies the whole layout of the individual Web pages; (ii) *Style Sheet* specifies the rendering of the different resources appearing in a Web page; (iii) *Custom View* is useful with all those resources, whose rendering is not performed by the two preceding modules.

The View essentially decouples the presentation of the individual resources from the layout structure of the Web pages in which they appear.

More specifically, the user interface is built automatically from the node of the graph being currently visited. Such interface displays information concerning the current resource and some further resources that are suitably related to the former. These latter resources are selected from those nodes of the graph of the Web application, that are connected to the current node through relations.

Controller. The *Controller* manages information processing. It is composed by the following modules: (i) the *Query Manager* is the module that interprets user requests (i.e. interaction with some Web application deployed within *Borè*) and translates them into the corresponding browsing-operations or update-operations on the graph; (ii) the *Access Control* is essentially a message dispatcher, that coordinates nearly all modules of the *Borè* platform and controls the access of users (or groups of users) to the individual resources; (iii) – each resource can be associated to one or more events – The *Event Manager* module handles the events through pluggable, customizable and extensible *Action Manager*s, which differ based on the nature of the events to notify to users. Some action managers should be provided by default (e.g., timers, email notifiers and so forth).

Model. The *Model* component is the data storage layer. It offers primitive functions both for query answering and for updating the graph of the Web application as well as the taxonomies of the relative resources and relations.

Resources and relations are separately managed by the *Resource Manager* and the *Relation Manager*, respectively. The Resource Manager (resp. Relation Manager) is responsible for accessing, storing, updating and versioning the individual resources (resp. relation) and the resource (resp. relation) taxonomies.

Resource and Relation Managers communicate with the *Storage Engine*, which handles row data. Row data is retrieved through the *Data Index*, which contains meta data type information, and the *Cartridge Manager*, which loads, stores and updates the various instances by means of pluggable, customizable and extensible storage units, for instance DBs, files, etc.

The *Profile Manager* module creates resource profiles by analyzing the requests for the individual resources.

Finally, the *Knowledge Discovery* module analyzes users' interactions with the resources of a Web application for the purpose of inferring suitable profiles. These are stored and managed by the *Profile Manager* module and allow Web application to provide customized support to the individual user. Users' profiles are learnt through the application of techniques from the fields of Web Usage Mining and Artificial Intelligence to their clickstream data, in order to discover suitable behavioral patterns, that ultimately allow the customization of the browsing experience. Moreover, further techniques from the areas of Collaborative Filtering are leveraged to analyze users' preference data and estimate their interest into not yet experienced resources. As it will be discussed in sec. 5, this allows the delivery of personalized suggestions to potentially interesting resources.

4 The *Borè* Query Language

The Query Manager module translates each user interaction (simple queries, i.e. click stream, and advanced queries, i.e. queries performed via a specific form) with the Web interface of the accessed Web application into a suitable mathematical set language, that is well suited to graph-based interpretation of the Web application itself. The language is the actual engine of the *Borè* platform and allows a high degree of modularity: each architectural module can be modified, reused or re-implemented without modifying other components. We next review some fundamental aspects of such language, by exploiting the definitions expressed in sec. 2.

The starting point is represented by some of its most basic operators. One such operator is IsA: given a type t and a taxonomy T, such that $t \in T$, IsA returns all the types which lie on the path from the taxonomy root to t. Formally:

$$IsA(t,T) = \{t\} \cup \{t'|\exists t'(t' \to t) \in T\} \cup \{t'|\exists t'' : (t'' \to t) \in T \wedge t' \in IsA(t'',T)\}$$

where $(x \to y)$ is a tree branch with x as the parent, while y is the child: in other words y inherits from x.

Another operator is *type*: given a resource (relation) instance i and a type taxonomy T, *type* returns t, the last type, top-down following the hierarchy in T, i belongs to.

$$t = type(i,T)$$

The *typeList* operator is closely related to IsA and *type*: given a resource (relation) instance i and RTT, *typeList* is defined as follows:

$$typeList(i,RTT) = IsA(type(i,RTT),RTT)$$

By building on the three basic operators discussed above, it is possible to define more sophisticated operators. For instance, $filterDown$ and $filterUp$ enable user browsing in the Web application. Given a resource r, $filterDown$ permits to select all neighbor resources $\{r_1,\ldots,r_n\}$ such that a direct edge from r to r_i (where $i = 1,\ldots,n$) exists in the Web application graph.

Formally, given a resource r, two subsets $RTT' \subseteq RTT$ and $ETT' \subseteq ETT$, and a boolean function δ (that represents some generic binary property):

$$\begin{aligned}filterDown(r,RTT',ETT',\delta,RTT,ETT,R,E) = \{r'|r,r' \in R \wedge (r \Rightarrow r') \in E \\ \wedge IsA((r \Rightarrow r'),ETT) \cap ETT' \neq \emptyset \\ \wedge typeList(r',RTT) \cap RTT' \neq \emptyset \wedge r' \vdash \delta\}\end{aligned}$$

where notation $(x \Rightarrow y)$ indicates a relation from the resource x to the resource y and $r' \vdash \delta$ means that δ holds true on r'. Symmetrically, $filterUp$ allows to navigate indirect edges in the graph:

$$\begin{aligned}filterUp(r,RTT',ETT',\delta,RTT,ETT,R,E) = \{r'|r,r' \in R \wedge (r' \Rightarrow r) \in E \\ \wedge IsA((r' \Rightarrow r),ETT) \cap ETT' \neq \emptyset \\ \wedge typeList(r',RTT) \cap RTT' \neq \emptyset \wedge r' \vdash \delta\}\end{aligned}$$

To better understand the transparent translation in *Borè* of user interactions with the Web front-end into as many corresponding queries (whose results collectively provide the new content of the Web front-end to supply to the user in response to the foresaid interactions), we next enumerate some example types of queries. These are meant for the example of fig. 1 and their role will be clarified in the study case of 6.

$$filterDown\ (\ \text{Un. of Calabria}, \{Object\}, \{Publishes\},$$
$$null, RTT, ETT, R, E) = \{\text{News 1}, \text{Event 1}\} \tag{1}$$

This query returns all resources, whose type lists contain *Object*, that are reachable from *Un. of Calabria* through a *Publishes* relation, assuming the taxonomies *RTT ETT R* and *E* and assuming there is no constraints (i.e. δ function is null).

$$filterDown\ (\ \text{Un. of Calabria}, \{News\}, \{Edge\},$$
$$null, RTT, ETT, R, E) = \{\text{News 1}\} \tag{2}$$

This query returns all resources, whose type lists contain *News*, that are reachable from *Un. of Calabria* through a *Edges* relation, assuming the taxonomies *RTT ETT R* and *E* and assuming there is no constraints (i.e. δ function is null).

$$filterUp\ (\ \text{Un. of Calabria}, \{Person\}, \{Collaborates\}, null,$$
$$RTT, ETT, R, E) = \{\text{Alice Moore}, \text{John Smith}, \text{Bob Jackson}\} \tag{3}$$

This query returns all resources, whose type lists contain *Person*, that reach *Un. of Calabria* through a *Collaborates* relation, assuming the taxonomies *RTT ETT R* and *E* and assuming there is no constraints (i.e. δ function is null).

$$filterUp\ (\ \text{Un. of Calabria}, \{Person\}, \{Collaborates\},$$
$$age < 30, RTT, ETT, R, E) = \{\text{John Smith}, \text{Bob Jackson}\} \tag{4}$$

This query returns all resources, whose type lists contain *Person*, that reach *Un. of Calabria* through a *Collaborates* relation, assuming the taxonomies *RTT ETT R* and *E* and assuming that we are looking for under-30 year old people.

$$filterUp\ (\ \text{Un. of Calabria}, \{Person\}, \{Works\},$$
$$age < 30, RTT, ETT, R, E) = \{\text{John Smith}\} \tag{5}$$

This query returns all resources, whose type lists contain *Person*, that reach *Un. of Calabria* through a *Works* relation, assuming the taxonomies *RTT ETT R* and *E* and assuming that we are looking for under-30 year old people.

5   Advanced Web3.0 Features of *Borè*

In this section we discuss some advanced Web3.0 features of the proposed *Borè* platform, i.e. the generation of social networks and the collaborative filtering.

Social Cooperations. One of the basic ideas of Borè is that the user should be free to create its own Web "universe" and share it with other users. The user is the center of the web: even the user is a Web resource.

Let us suppose that a user creates several Web applications and publishes them to other users. An environment of social cooperation and social networking spontaneously arises if such applications share resources, since different users interact with one another.

Fig. 3(a) shows a scenario in which there are two communities. Each community has a set of resources (such a set is called *Resource Pool* in fig. 3(a)) and some *Shared Resources*. Both the communities are allowed to read, possibly modify the shared resources and can be notified on any updates to them.

To elucidate, in the context of the toy example of fig 1, the two communities can be *University of Calabria* and *Alice Moore*. Hypothetically, we can imagine that the only shared resource between the two communities is *News 1*. The latter essentially enables a social cooperation between *University of Calabria* and *Alice Moore*: a detailed explanation is in sec. 6.

The social nature of the proposed architecture calls for a flexible management of privileges on the resources. We propose a Unix-like privilege system, where each file is assigned to an owner and a group. The difference is that in our scenario a node could be assigned to several communities, so we chose to associate to each node of the graph a rich set of privileges.

A user, for instance, can access or edit a node in the graph because he/she is the owner or because he/she is a member of the group which created the resource.

Formally, a privilege G is a tuple $G = \{\alpha, \beta, \gamma\}$, where: α is the community the resource belongs to; β is a constraint over α: G can be assigned to the sub-set of α, for which β holds; γ is the privilege type: it is the set of all actions allowed on the resource.

Each resource can have 0, 1 or more privileges, which attribute a series of available actions on the resource to a subset of users in a community. This is a very fine-grained privilege management (on a single node). In many practical cases, however, this approach might be too complex and impractical, especially in configurations with a large number of nodes. For this reason a coarse-grained configuration is appreciated. In fact, it allows to select and assign privileges for a group of nodes with the same type. For example, a company might decide to public (make visible to guests) all the news, but to reserve other types (such as invoices) only for the members of some communities and/or their owners. This approach significantly simplifies the management of privileges, but at the same time decreases assignment flexibility. A mixed approach is, instead, in our opinion the optimal strategy. A set of coarse-grained privileges can be assigned and, where necessary it is possible to define fine-grained privileges. By turning to the previous example, the company could prohibit to any guest user the invoice querying (coarse-grained constraint), but it can give to a guest user the possibility to see his own invoices (fine-grained privileges).
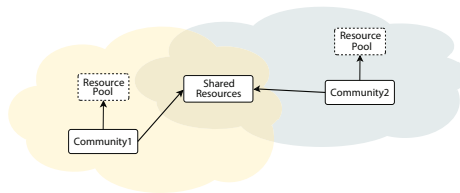
Collaborative Filtering. The social cooperation environment leads to a really interesting aspect: a lot of information is shared among the users. This information can be used to activate knowledge discovery processes (within the

Knowledge Discovery module) in order to infer new knowledge that is used to enrich user browsing experience.

The collaborative filtering process in Borè consists of the following steps: (i) some relevant relations, among resources, are identified: recall that resources comprise individuals and communities; (ii) for each relation, a correspondence matrix is built. The number of rows in the matrix corresponds to the resources (i.e., individuals or communities), while the columns of the matrix represent all the resources involved in the chosen relation; (iii) suitable collaborative filtering techniques [1,9] are applied on the formed matrix; (iv) finally, new relations and activities with resources, according to the discovered patterns at the previous step, are suggested to the end user.

Collaborative filtering techniques aim to find a social neighborhood (i.e. a community) of the end user, i.e., other users with similar preferences who have experienced resources not yet known to the end user. Such resources represent potentially interesting suggestions. Once the community to which the end user belongs to is identified, those resources that are expected to be mostly interesting to the her/him are recommended.

To clarify, let us consider the example relation *Collaborates* in fig. 3. The relative matrix is shown in fig. 3(b). Let us suppose that the two users r_i and r_j belong to the same preference community; in our example, both of them collaborated with r_a, r_c, r_e and r_f. Since r_i and r_j share similar tastes (they belong to the same community), and r_j also likes to collaborate r_h, probably r_i might like to collaborate with r_h too. For this reason, r_i is suggested r_h.

(a) Social Cooperation

"Collaborates" Relation

	r_a	r_b	r_c	r_d	r_e	r_f	r_g	r_h	r_i
...									
r_i	yes		yes		yes	yes			
r_j	yes		yes		yes	yes		yes	

(b) Collaborative Filtering Data Matrix

Fig. 3. Social Cooperations and Collaborative Filtering

6   A Case Study

In this section we present a case study consisting of some screenshots of the Web application introduced in the toy example of fig. 1. The case study is aimed to

highlight some major features of *Borè*. Let's suppose that the prof. *Alice Moore*, one of the workers of the institute *University of Calabria*, wishes to call a Faculty Council, provided that she has the necessary privileges.

The first step is the accessing the home page of the institute (see fig. 4(a)). The output of *Borè* is essentially a collection of frames, each of which corresponds to a specific graphical representation of the Web contents. The main frame shows a description of a resource, whose type is *Institute* and whose realization is *University of Calabria*. The description shows the values of the fields of such a resource, i.e., *Name* and *Activity Description*. The side boxes contain the results of the *filterUp* and *filterDown* operators (introduced in sec. 4) and of the *Knowledge Discovery* module. In particular, the top frame (i.e., the *Forward Link Box*) contains the answer to the query 1 in sec. 4. The middle frame (i.e., the *Backward Link*) reports the answer to the query 3 in sec. 4. The bottom box reports the recommendation list constructed by the *Knowledge Discovery* module on behalf of the end user.

When *Alice* follows the *Alice Moore* link, her personal home page is displayed (see fig. 4(b)). The latter is again composed of a main frame that displays her personal information as well as a set of side boxes. The filter-up and filter-down queries to fill such side boxes are automatically issued within *Borè* when *Alice* moves from the Web page of *University of Calabria* (in fig. 4(a)) to her personal Web page (in fig. 4(b)). Notice that, in the graph model of the toy example of fig. 1, the resource *Alice Moore* has no incoming edges. Therefore, the filter-up operator returns an empty set and, hence, the backward link box is not visualized at all on the personal Web page. Additionally, the box named *Shared* contains all resources that *Alice Moore* shares in the Web application with *University of Calabria*. The *Shared* box actually connects *Alice Moore* to her social network.

Since *Alice* wants to call a *Faculty Council*, she has to create a new resource type, and then create an instance of the new defined type. A form for the definition of a new type is shown in fig. 4(c). On the left side there is a tree representation of the resource type taxonomy: once selected the desired type, the new type definition will be appended in the hierarchy. On the right side there are tools for the definition of the fields of the new type.

The form for resource instantiation is shown in fig. 4(d). Such a form is used by *Alice* to create a resource (i.e., an instance) of the type *Faculty Council* by providing values for each field of the type. *Alice* can also specify a connection between herself and the new resource that specifies the type of relation between them (such a relation appears in the graph model of 1).

Among the other operations, Alice can send an invitation to other people or, even, associate a reminder that will dispatch emails to the participants a day before the Faculty Council. In *Borè*, these operations correspond, respectively, to the addition of a new relation, say *Participates* (connecting the resources corresponding to the invited people and the council) and to the association of a custom *Action Manager* to the newly created resource *Faculty Council*.
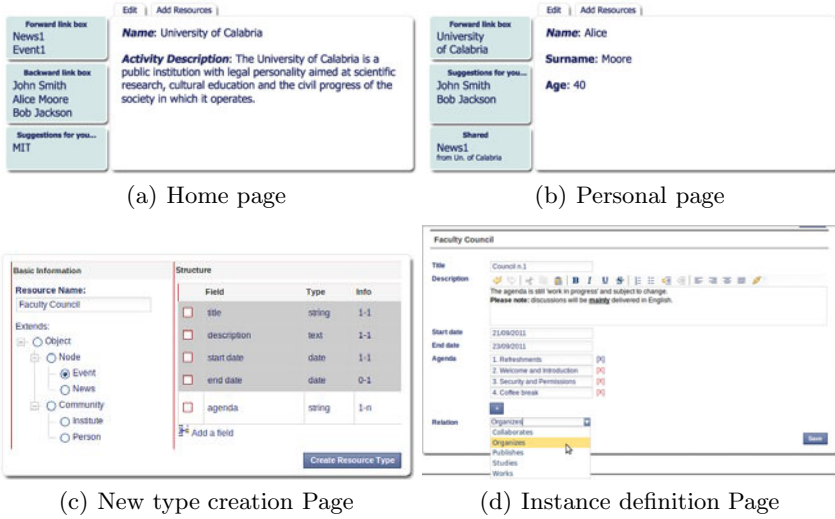
(a) Home page   (b) Personal page

(c) New type creation Page   (d) Instance definition Page

Fig. 4. Browsing and definition and instantiation of a new resource in Borè

7   Related Work

In this section, we discuss the main differences between *Borè* and some established competitors, namely frameworks and applications. Frameworks include Web infrastructures such as Django[4] and Ruby on Rails [5]. These are meant to support expert users in the development of applications based on the Model-View-Controller pattern. Applications are instead content-management systems, such as Joomla! [6] and Drupal [7], whose goal is allow (even inexpert) users to simply publish their contents on the Web through a wide variety of templates, components and tools. *Borè* exhibit meaningful differences with respect to both frameworks and applications.

In particular, one general limitation of frameworks is that these are not easily customizable and extensible. As a matter of fact, the definition of new types and extensions is usually a complex process that necessarily involves expert developers. For instance, the definition of new resource types in Django involves a non trivial coding process. Moreover, the graphical tools for resource instantiation and manipulation are accessible only to system administrators. Additionally, apart from the back-office functionalities, there is no default representation of the newly defined objects for the end user. By contrast, *Borè* allows the definition of new resources and relations through user-friendly embedded graphical tools (see sec. 6). Also, *Borè* offers a suitable interface for type instantiation and instance manipulation, that can be simply accessed and used by both users

---

[4] https://www.djangoproject.com/
[5] http://rubyonrails.org/
[6] http://www.joomla.org/
[7] http://drupal.org/

and system administrators. Furthermore, *Borè* is easily extensible through the implementation of well-defined interfaces.

8   Conclusions and Future Work

In this paper, we have provided further contributions in the context of the research we are conducting within *Borè*, a new architectural paradigm for developing content-based web applications based on Web3.0 principles. In particular, based on our previous contribution [2], we have detailed several advanced Web3.0 features of *Borè*, still recalling its principles and functional architecture. Future work is actually oriented in assessing the performance of *Borè* under two different settings. The first one refers to the effort the end-user need to pay in order to designing and personalizing Web3.0-enabled *Borè*'s views over the Semantic Web. The second one is instead pertinent to the issue of effectively and efficiently managing taxonomies underlying the *Borè* query language on a suitable DBMS layer.

References

1. Bell, R., Koren, Y., Volinsky, C.: Modeling relationships at multiple scales to improve accuracy of large recommender systems. In: KDD 2007: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 95–104. ACM, New York (2007)
2. Bevacqua, A., Carnuccio, M., Ortale, R., Ritacco, E.: A new architectural paradigm for content-based web applications: Borè. In: Proc. of Int. Symp. on Database Engineering and Applications (IDEAS), pp. 192–196 (2011)
3. Bondy, A., Murty, U.S.R.: Graph Theory, 3rd Corrected Printing. Springer (2008)
4. Cannataro, M., Cuzzocrea, A., Mastroianni, C., Ortale, R., Pugliese, A.: Modeling Adaptive Hypermedia with an Object-Oriented Approach and XML. In: Proc. of Int. Ws. on Web Dynamics, WebDyn (2002)
5. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. Morgan-Kaufmann (2002)
6. Cesario, E., Folino, F., Ortale, R.: Putting Enhanced Hypermedia Personalization into Practice via Web Mining. In: Galindo, F., Takizawa, M., Traunmüller, R. (eds.) DEXA 2004. LNCS, vol. 3180, pp. 947–956. Springer, Heidelberg (2004)
7. Fay, C., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A Distributed Storage System for Structured Data. Google (retrieved November 13, 2009)
8. Hamilton, J.: Perspectives: One Size Does Not Fit All (retrieved November 13, 2009)
9. Marlin, B.: Modeling user rating profiles for collaborative filtering. In: NIPS*17 (2003)
10. Mitchell, J.: 10 "Concepts in object-oriented languages". Concepts in programming language, p. 287. Cambridge University Press, Cambridge (2002)
11. Peters, I., Becker, P.: Folksonomies. Indexing and Retrieval in Web 2.0. Knowledge & Information: Studies in Information Science 2009. De Gruyter/Saur (2009)
12. Scott, M.L.: Programming language pragmatics, 2nd edn., p. 470 vikas, Morgan Kaufmann (2006)

Adaptive Topic Community Tracking in Social Network

Zheng Liang, Yan Jia, and Bin Zhou

Institute of Software, Department of Computer, National University of Defense Technology,
Changsha 410073, China
zliang@nudt.edu.cn, jiayanjy@vip.sina.com,
bin.zhou.cn@gmail.com

Abstract. Large social networks (e.g. Twitter, Digg and LinkedIn), have successfully facilitated information diffusion related to various topics. Typically, each topic discussed in these networks is associated with a group of members who have generated content on it and these users form a topic community. Tracking topic community is of much importance to predict the trend of hot spots and public opinion. In this paper, we formally define the problem of topic community tracking as a two-step task, including topic interest modeling and topic evolution mining. We proposed a topic community tracking model to model user's interest on a topic which is based on random walk algorithm and combines user's personal affinity and social influence. And then, considering that user's interest on a topic will vary with time when the discussion content changes and new topic community member joins in, we explore an Adaptive Topic Community Tracking Model. Comprehensive experimental studies on Digg and Sina Weibo corpus show that our approach outperforms existing ones and well matches the practice.

Keywords: Topic Community, Personal Affinity, Social Influence, Random Walk, Social Network, Algorithm.

1   Introduction

In the Web 2.0 age, Information Diffusion in large social networks (such as Twitter[1], Digg [2]and Sina Weibo[3]) is of much importance in tracking public opinion, launching new products, and other applications. Within these social websites, individual users are allowed to publish their posts or comments to diffuse certain topics. Fig 1 depicts the structure and evolution of such diffusion network. The ellipses represent different users (e.g., U1, U2, U3, U4, U5, and U6), and each ellipse contains users' posts or comments. The edges in the figure indicate hyperlinks between posts. Observe that for a particular topic such as topic "IPhone", which is decrypted with key words in the circle, there is a group of users who have published their comments or posted on it and these users form a community. In the sequel, we refer to such a community as

---

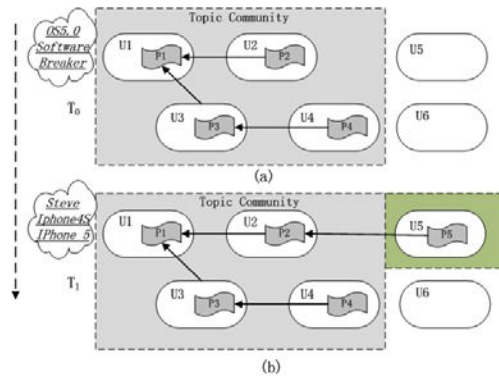[1] http://twitter.com

[2] http://digg.com

[3] http://www.weibo.com

Fig. 1. Topic Community Tracking for topic "IPhone"

topic community. Clearly, these communities are a potential gold mine for hotspot trend and market analysts.

In this work, we explore the problem of topic community tracking for information diffusion. For example, consider the evolution of network structure and topic content within the topic community on "IPhone" shown as Fig. 1. In time window T_0, user U1 wrote post P1 and starts a new discussion on "OS5.0 software breaker". Later, user U2 read post P1 and joined the topic community by writing post P2 and explicitly creating a hyperlink to P1. Similarly, new users U3 and U4 join the topic community by writing posts linking to existing posts. The modified structure of topic community in time window T_1 is depicted in Fig. 1(b). Notice that U5 has joined the topic community and change the discussion content to "IPhone 4S and IPhone 5". For this phenomenon, this paper focus on three central questions as followed.

1) What factors influence users to join topic community?
2) How can we make use of various factors such as topology and content information to predict community member in future?
3) Do the temporal features such as topic content evolution affect the users' behavior along time?

To answer the questions above, we have to analysis human behaviors. According to the behavior theory, human's actions happen by both internal and external cause [5]. That is, a user tends to post comments in response to social influence, or in response to personal affinity. Moreover, the evolution of topic content can also affect the topic diffusion behavior.

Recently, quite a few related studies have been conducted, for instance, information diffusion [1, 4], community detection [10, 11] and social influence [2, 3]. However, topic community tracking problem addressed in this paper is very different from these works. Majority of research on information diffusion paid their attention on the macroscopic level, such as degree distributions, diameter, clustering coefficient [4, 12], while topic community tracking focus on the microscopic analysis on topic diffusion behavior. Community detection [10] is trying to split a gigantic network into several relatively independent parts, but not focuses on community evolution for

information diffusion. Although some work of Social influence [2] has exploited content to quantify the strength of the influence, but they do not consider the evolution of topic content. In topic community tracking problem, we try to simultaneously incorporating topology, content, and time factor simultaneously into a unified model.

Tracking topic community is useful in many applications. It not only facilitates the design of advanced social network system with more sophisticated personalized recommendations and filters, but also helps us to set up intelligent strategies in influencing the population faster by accelerating the information propagation process. In summary, our contributions are listed as follows.

First, we model personal affinity by investigating the content semantic similarity between user's historical publication and current topic model. Probabilistic Latent Semantic Analysis (PLSA) [7] is introduced to measure the similarity.

Second, we formulate the task of topic community tracking into an interest ranking problem. Moreover, topic evolution is also considered and an Adaptive Topic Community Tracking model (ATCT) is proposed to estimate community evolution for text stream data.

Finally, we test the proposed model on the two different data sets. The experimental results indicate that our analysis is useful and interesting, and our methods are effective and efficient.

The rest of the paper is organized as follows. Section 2 definitions the problem and overviews our methods. Section 3 introduces topic community tracking model based on random walk in detail. Section 4 describes the experiments conducted on real world data sets to validate the effectiveness of the proposed methods. Finally, Section 5 discusses the related work and Section 6 concludes and discusses future work.

2   Overview

2.1   Definitions

Let us begin with several necessary concepts in this paper.

Definition 1 Document. Let $D = \{d_1, d_2, \ldots, d_M\}$ be the documents collections in social network, where document $d_i \in D$ is the text document(s) published by user u_i and M is the number of user set. Document d_i consists of Reply-Document and Self-Document. Let R_{ij} be the Reply-Document consisting of replies from user u_j to u_i. Besides, we use S_i to denote the Self-Document consisting of posts which are not replies to any user. To consider the time factor, we further denote $D_{i,k}, S_{i,k}, R_{ij,k} (k \in [1, N])$ as the documents, self-documents and reply-documents created at time window t_k respectively.

Definition 2 Topic Community. A topic community is a group of users who discuss one specific topic by posting or replying topic documents D^E. We denote TC^E as Topic-Community of topic E and let TC_k^E be the users joining the TC_k^E in time window t_k.

Definition 3 Topic Interest. For a given Topic E, user u_i has a certain level of interest in joining Topic Community at time window t_k. Topic Interest $h_{i,k}^E \in H_k^E$ reflects the probability of user u_j joining TC_k^E at time window t_k.

2.2 Intuition

Before formally introducing the model, we first analyze several key correlations that motivate the model.

Correlation 1 Topic Interest and Social Influence. Researches on social influence show that a person's interest may be influenced by his friends. Several key factors reflect the influence strength, such as the frequency of communication. Moreover, social influences are associated with different topics [2, 13]. So we define Topic Influence $fh_{ij}^E \in FH^E$ to be social influence from user u_i to u_j on topic E. In most cases, the social influence score is asymmetric, i.e. $fh_{ij}^E \neq fh_{ji}^E$.

Correlation 2 Topic Interest and Historical Behavior. It has been shown in human behavior that more frequently user comment or cite the posts on a certain topic, more interesting users have which generated by users' research or hobby. So we defined the part of topic interest from personal affinity as $sh_{i,k}^E \in SH_k^E$.

Correlation 3 Topic Interest and Topic Evolution. Users' topic-Interest may change over time with topic content. In addition, the documents in the text stream created in the same time stamp will share similar semantic background. That means the topic feature vector at time window θ_k^E would associated with the content θ_{k-1}^E.

3 Adaptive Topic Community Tracking Model

In this section, we propose an Adaptive Topic Community Tracking (ATCT) to capture personal affinity, social influence and topic evolution simultaneously.

3.1 The General Model

At time window t_k, we already know users' historical documents $D_{1,\dots,k-1}$ and topic documents D_k^E, Let us assume that we've already known the previous topic feature vector θ_{k-1}^E, we want to infer the topic interest value H_k^E and topic θ_k^E and predict users' topic diffusion behavior at future period.

As discussed in Section 2.2, the topic interest of an individual depends on personal affinity and social Influence. On one hand, topic influence FH_k^E is associated with Reply-Documents $R_{1,\dots,k-1} \in D_{1,\dots,k-1}$ and topic feature vector θ_k^E at current time window based on Correlation 1. On the other hand, Personal Affinity can be estimated from his/her historical Self-Documents $C_{1,\dots,k-1} \in D_{1,\dots,k-1}$ and topic feature tor θ_k^E. Based on Correlation 3, given topic documents D_k^E and topic content θ_{k-1}^E, the current topic feature vector θ_k^E depends on the current status and its contextual. With the above analysis, the framework of our model can be represented as Eq. (1).

$$P\left(H_k^E, \theta_k^E \middle| D_k^E, \theta_{k-1}^E, D_{1,\dots,k-1}\right) = P\left(H_k^E \middle| D_{1,\dots,k-1}, \theta_k^E\right) P(\theta_k^E | \theta_{k-1}^E, D_k^E)$$

$$= \left[P\left(SH_k \middle| C_{1,\dots,k-1}, \theta_k^E\right) \diamond P\left(FH_k \middle| R_{1,\dots,k-1}, \theta_k^E\right)\right] P(\theta_k | \theta_{k-1}^E, D_k^E) \tag{1}$$

In Eq. (1), the first component $P(H_k|D_{1,\dots,k-1}, \theta_k)$ is topic interest model and the second component $P(\theta_k^E|\theta_{k-1}^E, D_k^E)$ is topic evolution model. In topic interest model, we denote the first component $P\left(SH_k \middle| C_{1,\dots,k-1}, \theta_k^E\right)$ as Personal Affinity model and the second component $P\left(FH_k \middle| R_{1,\dots,k-1}, \theta_k^E\right)$ as Social Influence Diffusion Model, which are described in 3.2 and 3.3. Besides, $\diamond$ between personal affinity and social influence model is the integration function which will be introduced in 3.4. In the topic interest model, a novel Topic Interest Ranking Algorithm is proposed to predict the behavior of joining topic community, which is denoted as static version Topic Community Tracking Model (TCT). Finally, we propose an Adaptive version Topic Community Tracking Model (ATCT) to integrate topic evolution model and topic interest model, which is described in 3.5.

3.2   Personal Affinity Model

To model personal affinity on topic E, we need to exploit the frequency of self-documents $C_{1,\dots,k-1}$ and its content similarity with topic feature vector θ_k^E. For example, in the topic community of "NBA Final Game", more self-documents and higher similarity between topic discussion and self-documents which a user u_i post, more passion the user u_i have to the topic.

Usually, documents can be represented with bag of words. However, using bag of words will raise a problem of high data dimensionality. As demonstrated in [6], when the data dimensionality is too high, the performance of cosine similarity will become worse. To solve this problem, we perform Probabilistic Latent Semantic Analysis (PLSA) [10] to reduce data dimensionality and to generate a latent themes space for representing content feature vector.

After representing each document as a feature vector, Cosine similarity [15] can be used to calculate the similarity between two content vectors. Subsequently, the personal affinity sh_i^E to topic E of user u_i is then calculated as in Eq. (2):

$$sh_i^E = \sum_{d \in C_i} Cosine(\theta^d, \theta^E) \tag{2}$$

where $Cosine(\theta^d, \theta^E)$ is the cosine distance based on PLSA model, which can be represented as:

$$Cosine(\theta^d, \theta^E) = \frac{\sum z^E z^d}{\sqrt{[\sum(z^E)^2][\sum(z^d)^2]}} \tag{3}$$

$z^E \in \theta^E$ is the latent themes generated for topic E with and $z^d \in \theta^d$ is the feature vector for users' documents correspondingly.

Finally, the normalized vector of Personal Affinity values $Q^E = \{q_1, q_2, \dots, q_M\}$ is given by Eq. (4)

$$q_i = \frac{sh_i^E}{\sum_{i=1}^{M} sh_i^E} \tag{4}$$

Personal Affinity vector Q^E shows how likely topic content "attracts" users to join in the topic discussion.

3.3   Social Influence Diffusion Model

Now we consider topic influence which shows how likely users influence each other in adoptions of a topic. The influence among individuals is different with their social tie [14], which is analogous to virus propagation problem of the epidemic research.

First, based on the historical behavior of reply relationship, we first construct the affinity Matrix $w_{ij} \in W_K (i, j \in [1, M])$ associated to the influential network G, where w_{ij} represent the influence strength user from u_i to u_j.

$$W = \begin{bmatrix} 0 & \cdots & w_{1M} \\ \vdots & \ddots & \vdots \\ w_{M1} & \cdots & 0 \end{bmatrix}$$

Next, we consider two key factors which affect the influence value w_{ij}. First of all, it is well recognized that social influence patterns regarding to different categories or topics. Given a special topic E, we define topic influence as Eq. (5).

$$w_{ij}^E \propto \sum_{d \in R_{ij}} Cosine(d, \theta^E) \tag{5}$$

Second, influence strength w_{ij} is proportional to the reply frequency of user u_j to u_i. To integrate these two important factors, the influence value w_{ij} associated to topic E is defined as Eq. (6), where $F(R_{ij})$ represents the frequency of each user u_i replied by u_j.

$$w_{ij}^E = F(R_{ij}) * Cosine(\theta^d, \theta^E) \tag{6}$$

In the 3.4 subsection, we generate the transition probabilities of topic influence based on $w_{ij}^E \in W_k^E$ and apply it to topic interest ranking algorithm.

3.4   Topic Community Tracking Model

Given Personal Affinity Vector Q_k^E and Topic Influence Matrix W_k^E, this subsection describes the integration function ⋄ which combines personal affinity model and social influence diffusion model for predicting the vertexes joining topic community in future. We address this problem as ranking problem and propose a novel Topic Interest Ranking Algorithm (*TIRank*), which is denoted as a static version Topic Community Tracking Model (TCT).

To integrate those models above, let us analyze two types of behaviors of joining the topic community at time window t_k firstly. As discussion in Section 2.2, the action of publishing self-documents is generated from personal affinity SH_k, while the action of publishing reply-documents is the result of topic influence FH_k. The topic

diffusion network consists of nodes, hidden nodes, and the links among them. One can imagine that there is a hidden node which links all the vertexes to contribute Personal Affinity to the network.

Therefore, we assume that a user takes self-publish action with the probability $(1 - $ ty $(1 - \beta) * SH$, while reply-publish action with $\beta * FH$. Hence, we define Topic Interest as the following Eq. (7):

$$TIR_i^E = \beta w_{ij}^E \sum_{j \in Nb(i)} TIR_j^E + (1 - \beta) q_i^E \tag{7}$$

The part $\beta w_{ij}^E \sum_{j \in Nb(i)} TIR_j^E$ represents the topic influence generated from the friends of u_i, while the other part $(1 - \beta) q_i^E$ represents personal affinity of u_i generated by users' own. The interest source balance parameter β is used to control the balance of topic influence and personal affinity.

TIRank is analogous to personalized PageRank [8] and corresponds to a problem of Random Walk with Restarts [9]. To achieve this ergodic Markov chain, the revised transition probability matrix $\overline{W_K^E}$ determining the random walk on G can be defined as

$$\overline{w_{ij}^E} = \begin{cases} \dfrac{w_{ij}^E}{\sum_{j=1}^{M} w_{ij}^E} & if \sum_{j=1}^{M} w_{ij}^E \neq 0 \\ \dfrac{1}{M} & if \sum_{j=1}^{M} w_{ij}^E = 0 \end{cases} \tag{8}$$

Moreover, due to the presence of dangling nodes which do not have any out-links and cyclic paths in the network, we apply the remedy of "random jumps" as what PageRank does. The normalized topic influence matrix W after the adjustment can be written as:

$$C_k^E = \alpha \overline{W_k^E} + (1 - \alpha) e e^T \tag{9}$$

where α is the probability that the random walk follows a link. When a Markov chain is ergodic, the stationary distribution of its Markov matrix is guaranteed to exist, so $TIR_k^E(t + 1)$ will converge to TIR_k^E. Substituting TIR_k^E for TIR_{t+1}^E and TIR_t^E, then following some algebraic steps, we obtain the ranking score by Eq. (10).

$$TIR_k^E = (1 - \beta)(I - \beta(C_k^E)^T)^{-1} Q_k^E \tag{10}$$

3.5   Adaptive Topic Community Tracking Model

Now we consider evolution component $P(\theta_k^E | \theta_{k-1}^E, D_k^E)$ to propose an Adaptive version Topic Community Tracking Model (ATCT) by incorporating time lapse factor for topic discussions.

As discussion in Correlation 3, Topic Interest will change with topic evolution over time. The example mentioned above illustrates that the recent topic documents D_k^E are relatively good indicator for the tendency of topic content, rather than that of long time ago. In addition, the documents in the text stream created in the same time stamp will share similar semantic background. Hence, we introduce time factor for updating topic vector adaptively as Eq. (11).

$$\theta_k^E = \{\theta_{D_k^E}^E + exp(-\gamma * \Delta t^E)\, \theta_{k-1}^E\}_{nomal} \tag{11}$$

In Eq. (11), $\theta_{D_k^E}^E$ is the content vector generated from topic documents D_k^E with PLSA, Δt^E represents how much time windows lapse from the publishing time of the topic document set D_{k-1}^E to current time (or time for prediction), and γ is a forgetting parameter that reflects how quickly topic content changes over time. $\{*\}_{nomal}$ means the normalize function. Now, Adaptive Topic Community Tracking Model is showed as Fig. 2.

Algorithm 1: Adaptive Topic Community Tracking Model

Input: Topic Document Stream segment D_k^E
         Reply-Documents Stream segment R_k
         Self-Documents Stream segment S_k
Output: Topic interest Rank (TIR_k^E) and θ_k^E
Begin
For $t_k == 1\ to\ N$ do
 If ($t_k == 1$) then
    1) Generate topic feature vector θ_1^E with PLSA model
    2) Generate Vector Q^E from Self-Documents Set S_1 by Eq. (4)
    3) Generate Topic Influence Matrix W_1^E from R_1 by Eq. (6).
    4) Compute Topic Interest Rank TIR_1^E by Eq. (10).
 Else
    5) Generate Topic documents vector $\theta_{D_k^E}^E$ from D_k^E with PLSA model
    6) Update θ_k^E with $\theta_{D_k^E}^E$ and θ_{k-1}^E by Eq. (11)
    7) Update W_k^E and Q_k^E with θ_k^E by Eq. (6).
    8) Update Topic Interest Rank TIR_k^E by Eq. (10).
End

Fig. 2. Adaptive Topic Community Tracking Model

4   Experiment

In this section we present our experimental results comprising a description of dataset, experiment setup, parameter settings and comparative study of our method.

4.1   Datasets

We perform our experiment on two different genres of dataset: Digg and Sina Weibo. Digg data set is crawled all stories in 51 popular Newsroom which are submitted from July to September 2011. In total, this dataset comprises 21,919 users, 187,277 stories, 687,616 comments and 477,320 replies. Another data set which crawled 6 month data from "Micro-Group" of Sina Weibo, contains 14 thousand users, 1.5 million posts and 3.3 million comments.

In our work, we consider that the discussions in each group belong to the same topic. In other words, all the posts in topic groups belong to the same topic.

4.2   Experiment Setup

To measure the performance of our model, we evaluate the accuracy of predicting topic community joining behavior of special group in future time window. The two datasets are partitioned into 12 continuous time windows and each time window is two weeks long. We assign one with 8 time windows into training for model turning and other with the left 4 time windows into test set for model validation.

We employ the precision at Top-K as evaluation metrics, i.e., how many of top K users estimated by our algorithm actually comment on or cite the new post.

4.3   Compared Predictive Methods

We compare the following methods for predicting topic diffusion behavior in future time window.

Method 1 Historical Information (HI). Method 1 is a basic form of probability based on user historical post number. The calculate method is show as:

$$P(u_i|\theta_k^E) = \frac{Num(D_{i,1\ldots k-1})}{\sum_{j=1}^{M} Num(D_{j.1\ldots,k-1})} \tag{12}$$

In Eq. (12), $Num(D_{i,1\ldots k-1})$ represents the document number of user u_i in the group before time window k.

Method 2 Adamic-Adar Coefficient (AAC). Adamic-Adar Coefficient is reported to be the most effective ways to topic community tracking by network topological feature. Please refer to [12] for details.

Method 3 Content Analysis (CA). This method only considers users' content feature with personal affinity sh_i^E. So the probability of user u_i to join topic community TC_k^E is estimated by Eq. (13).

$$P(u_i|\theta_k^E) = \frac{sh_{i,1\ldots k-1}^E}{\sum_{i=1}^{M} sh_{i,1\ldots,k-1}^E} \tag{13}$$

4.4   Parameter Setting

This subsection evaluates the choice of parameters in Adaptive Topic Community Tracking model. Usually, the damping factor α is set to 0.85 and the number of latent topic can be determined by the Bayesian model selection principle [16]. The remaining parameters are fixed by tuning them on the training data. In this subsection, we choose four representative groups of Sina Weibo to estimate the parameters in our model and the setting of Digg is similar.

Choice of Interest Source Balance β

When evaluating on real data we notice that β plays an important role in topic community tracking model. Fig.3 (b) illustrates the significant difference in performance in tuning β. It can be observed that the value β is determined by the topic attribute of

the newsroom. Take the groups of Sina Weibo data as example, it depicts $\beta = 0.3$ perform best in group "Game", while a relatively small value $\beta = 0.1$ in group "IPhone" achieves good performance. This finding is understandable because that group "Game" related to hobbies team more strong ties between users than group "IPhone", where various customers gather together. Therefore, users in "Game" are more likely to be influenced by friends than that in "IPhone".

Choice of Topic Evolution Parameter γ

The result of turning Topic Evolution Parameter γ is illustrated in Fig. 3(b). We find the best choice of γ is determined by the topic attribute. The affection of γ in group "NBA" is more than group 'Canon'. A possible explanation for this is that "NBA" is so quick that fans only focus the news recently and "forget" the historical events, while the discussion of topic "Canon" evolves slower relatively and make semantic background to be more important in the model.

(a)   (b)
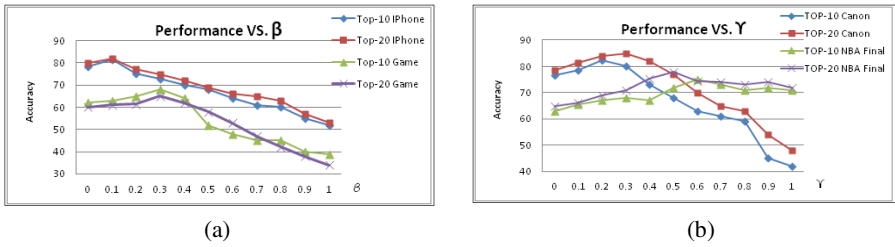
Fig. 3. (a) Performance vs. β (b) Performance VS. τ

4.5   Adaptive Prediction

Based on the exploration of parameter settings in the previous subsection, we focus on the prediction of topic community joining behavior.

Analysis on Topic evolution and Topic Interest Rank

Table 1 lists the highest probability words of group "IPhone" with PLSA model trained on the 14 thousands users of Digg. The quoted titles are our own interpretation

Table 1. Topic Evolution and Topic Interest Rank

Topic on 'IPhone'		
July 3rd	July 6th	July 9th
OS 5.0	**Steve**	**IPhone4S**
Software	**CEO**	**IPhone 5**
IPhone	Software	Software
Steve	OS 5.0	Steve
IPhone 5	IPhone4S	OS 5.0
The authors with highest topic interest rank score		
Fu2010	**Sa**	**Cheng**
Song	Bei	Song
Sa	Fu2010	Zai

of a summary for the topics. Blow the topic-word distribution are authors who are the highest probability to join the topic community in the groups.

As shown in Table1, topic content evolutes along time. For instance, users began to talk about the release of "OS5.0" which is the new operation system of "IPhone" on July 3rd. In the day of July 6th when Steve Jobs announced his resignation as CEO of Apple Company, the words "Steve" and "CEO" start to be on the list.

Besides the evolution of topic content, the results in Table 1 also suggest that the interest rank changes with the topic evolution. For example, "Fu2010", the top user on July 3rd, became activity obviously when the group focuses the 'OS5.0'. With the further investigation, we found user "Fu2010" is a programmer for apple software application. The same phenomenon can be found on the other time stamps, such as the period between July 6th and July 9th. In this time window, user 'Sa' on the list is a fanatic fan of "Steve", while "Cheng" is apple product fans.

According to the analysis above, our model tracks content evolution for popular events successfully and generates meaningful interest rank in social networks.

Performance Comparison

Here we address the problem of whether it is possible to infer who will possibly send a comment by communication records and the content of post-comment behavior.
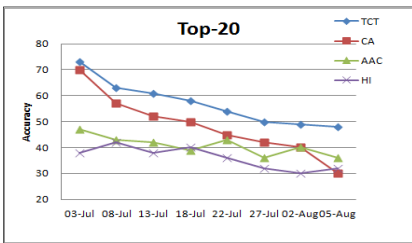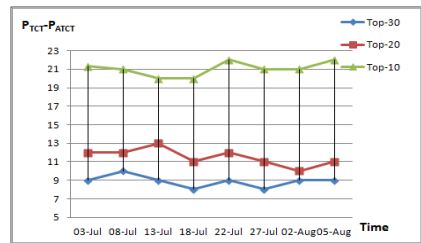
(a)   (b)

Fig. 4. (a) Top-20 VS. Time (b) Comparison between TCT and ATCT

In Fig. 4(a), we illustrate the prediction performance by comparing the HI, AAC, CA and the proposed static version Topic Community Tracking model (TCT). The result shows that the TCT model beats CA by 10% on accuracy. Moreover, the CA is better than AAC because that the importance of topic-interest is more significant than friend-interest. Furthermore the HI method is worst. Moreover, we obtain strong evidence of this hypothesis by observing that the performance of these models mono-tonically decays, which induces the models built at a specific time have decreasing predicting capability over time. This also implies our models well match the practice.

Moreover, the performance difference between TCT and ATCT with time and top-k is illustrated by Fig. 4(b). The vertical axis represents the performance gap be-tween TCT and ATCT, and the horizontal axis is time. The graphs show that, the gap between ATCT and TCT enlarges with increasing K values and the improvement gains are 43% and 53%, respectively. As a result, we obtain strong evidence of great-ly improvement by mining content evolution in ATCT models.

To sum up, by comprehensively considering historic, textual and structured infor-mation into a unified model, ATCT generates more accurate trends in social network.

5   Conclusions

In this paper, we propose a new way to automatic track topic community. Our algorithm is developed based on the intuition that a user's inclination to join topic discussion is motivated by personal affinity in topic content and social influence. First, PLSA is defended to analysis Personal Affinity by users' historical behavior. Moreover, we propose a topic interest ranking algorithm, which ranks users according to how important they are to predict the topic community joining behavior in future. The experimental results indicate that our analysis is useful and interesting, and our methods are effective and efficient.

References

1. Cha, M., Mislove, A., Grummadi, K.P.: A measurement-driven analysis of information propagation in the flickr social network. In: ACM WWW (2009)
2. Liu, L., Tang, J., Han, J., Jiang, M., Yang, S.: Mining topic-level influence in heterogeneous networks. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, pp. 199–208. ACM (2010)
3. Agarwal, N., Liu, H., Tang, L., Yu, P.: Identifying the influential bloggers in a community. In: Proceedings of the International Conference on Web Search and Web Data Mining, pp. 207–218. ACM (2008)
4. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: SIGCOMM 1999, pp. 251–262 (1999)
5. Dretske, F.: Explaining behavior: Reasons in a world of causes. The MIT Press (1991)
6. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The authortopic model for authors and documents. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, pp. 487–494. AUAI Press (2004)
7. Hofmann, T.: Probabilistic latent semantic analysis. In: Proc. of Uncertainty in Artificial Intelligence, UAI 1999, p. 21. Citeseer (1999)
8. Langville, A., Meyer, C.: Deeper inside pagerank. Internet Mathematics 1(3), 335–380 (2004)
9. Lovasz, L.: Random walks on graphs: A survey. Combinatorics, Paul Erdos is Eighty 2(1), 1–46 (1993)
10. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Physical Review E 69 (2004)
11. Tseng, B.L., Tatemura, J., Wu, Y.: Tomographic Clustering To Visualize Blog Communities as Mountain Views, Chiba, Japan, May 10-14 (2005)
12. Adamic, L., Adar, E.: Friends and neighbors on the web. Social Networks 25(3), 211–230 (2003)
13. Steyvers, M., Griffiths, T.: Probabilistic topic models. Handbook of latent semantic analysis, vol. 427 (2007)
14. Gilbert, E., Karahalios, K.: Predicting tie strength with social media. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems, pp. 211–220. ACM (2009)
15. Tan, P.N., Steinbach, M., Kumar, V., et al.: Introduction to data mining. Pearson Addison Wesley, Boston (2006)

Extracting Difference Information
from Multilingual Wikipedia

Yuya Fujiwara[1], Yu Suzuki[2], Yukio Konishi[1], and Akiyo Nadamoto[1]

[1] Konan University, 8-9-1 Okamoto, Higashi-Nada, Kobe, Hyogo 6588501, Japan
[2] Nagoya University, Furo, Chikusa, Nagoya, Aichi 4648601, Japan

Abstract. Wikipedia articles for a particular topic are written in many languages. When we select two articles which are about a single topic but which are written in different languages, the contents of these two articles are expected to be identical because of the Wikipedia policy. However, these contents are actually different, especially topics related to culture. In this paper, we propose a system to extract different Wikipedia information between that shown for Japan and that of other countries. An important technical problem is how to extract comparison target articles of Wikipedia. A Wikipedia article is written in different languages, with their respective linguistic structures. For example, "Cricket" is an important part of English culture, but the Japanese Wikipedia article related to cricket is too simple. Actually, it is only a single page. In contrast, the English version is substantial. It includes multiple pages. For that reason, we must consider which articles can be reasonably compared. Subsequently, we extract comparison target articles of Wikipedia based on a link graph and article structure. We implement our proposed method, and confirm the accuracy of difference extraction methods.

1 Introduction

Wikipedia[1], a large encyclopedia that is accessible using the Internet, is widely used throughout the world. An important policy of Wikipedia is that the contents of articles be the same for all language versions, which means that if we see any language version of an article of the same topic, the contents of the articles are expected to be exactly identical except for language. However, in fact, this policy is not obeyed, especially for topics related to culture. For example, the contents of an article about "Fish and Chips" are very rich in the English version, but they are very poor in the Japanese version because fish and chips is a very popular dish in the U.K., but not in Japan. We can infer that almost all articles written in Japanese are produced by Japanese native speakers, and that they do not know about fish and chips in any great detail. In stark contrast, U.K. residents are likely to know about fish and chips quite well, and that they can edit an article about fish and chips with some authority. Consequently, the contents of articles differ among respective language versions.

---

[1] http://en.wikipedia.org/

Generally, users browse the Wikipedia of their own native language; they would rarely refer to the other language versions because almost all users understand one language. The problem is that when users browse articles written in one language version, they cannot get information that is written in other language versions. In such cases, when users browse articles not only of their native version but also of other language versions, they can obtain more information. We propose a system to solve this problem: it displays not only articles written in the user's native language but also articles written in non-native languages.

Our proposed system compares Wikipedia articles written in the user's native language with Wikipedia articles of other languages written on the same topic. An important technical problem is how to extract comparison target articles of Wikipedia. A Wikipedia article is written in different languages, with their respective linguistic structures. For example, "Cricket" is an important part of English culture, but the Japanese Wikipedia article related to cricket is too simple. Actually, it is only a single page. In contrast, the English version is substantial. It includes multiple pages. For that reason, we must consider which articles can be reasonably compared. Subsequently, we extract comparison target articles of Wikipedia based on a link graph and article structure.

An important assumption underlying this system is that almost all the users of Japanese Wikipedia are Japanese native speakers, and that they understand English only a little. For that reason, the users need not only a Japanese Wikipedia article but also the other versions of Wikipedia articles. The process used for our proposed system is as follows.

1. Users input keywords in Japanese to the system.
2. The system retrieves one article related to keywords from the Japanese version of Wikipedia.
3. It retrieves articles related to keywords from the English version of Wikipedia.
4. It divides these articles written in Japanese and English into sections.
5. It compares sections in Japanese article and those in English articles, and detects which sections appeared in English articles but not in Japanese articles.
6. Then the system outputs Japanese article with sections of English articles that do not appear in the Japanese article.

The remainder of this paper is organized as explained below. First, in section 2, we summarize related work about extracting difference information. In section 3, we describe how to extract difference information. In section 4, we present discussion of our prototype system, experimental evaluation, and evaluation results. Finally, in section 5, we close with conclusions and future work.

2   Related Work

David et al.[5,6] and Michael et al.[9] extract measures of semantic relatedness of terms using the Wikipedia link structure. Takahashi et al.[10] propose a method to assess the significance of historical entities (people, events, etc.),

which uses the Wikipedia link structure to indicate how a historical subject affected other historical entities. Their proposed algorithm propagates such initial temporo-spatial information through links in a similar manner to that used for PageRank[2]. Nakatani et al.[7] propose a method of ranking search results adaptively by considering a user's comprehension level. In their method, the comprehensibility of search results is computed using the readability index and technical terms extracted from Wikipedia. They extract technical terms related to a search query by analyzing the link and category structure of Wikipedia. Jaap et al. [4] use Web inlinks for importance research related to the difference between Wikipedia and the Web link structure. Our research differs in that we seek to assess the difference between Wikipedia articles.

There are also research efforts that have been undertaken to build a thesaurus using the Wikipedia link structure. Nakayama et al. [8] proposed a method to analyze the Wikipedia link structure and produce a thesaurus dictionary. In the study, they use $pfibf$ as a term-weighting scheme and calculate the relevance between two Wikipedia articles. In fact, $pfibf$ is calculated using the number of hyperlinks to articles and the lengths of all paths in linked graph of Wikipedia articles. Chen et al.[3] propose a novel approach to construct a domain-specific thesaurus from the Web automatically using link structure information. Particularly, they form many tree called subtrees from the hierarchical structure of a Web site. Then they calculate the relevance of terms to use co-occurrence of words in trees. In contrast, we propose a method to use the term appearance position in the structures of articles. We use the number of interactive (out-links and in-links) to other articles to calculate relevance.

Some studies described in the literature make comparisons between languages using multilingual Wikipedia. Eytan et al.[1] extract the difference between the Wikipedia infoboxes that exist for other languages. Infoboxes are tables including summary data that exist for many Wikipedia articles. However, when comparing Wikipedia articles, specifically examining the structure of the table of contents can be superior in terms of the article contents when the article contents are not known.

3 Extraction of Difference Information from Articles

3.1 Extract Relevant Articles from English Wikipedia

In this section, we describe step 3 of the overview; extraction of English articles from Wikipedia. When we compare Japanese Wikipedia with English Wikipedia on a single particular topic, their granularity of information differs between the languages. For example, articles about "Cricket" exist in both Japanese and English Wikipedia. Articles on cricket in the Japanese version are written as descriptions of the rules and batting on a single page. However, the article related to cricket in the English version is written to include information on many related subjects in multiple pages, with great detail. As that example illustrates, it is necessary to consider the coverage of comparison articles. When a user's
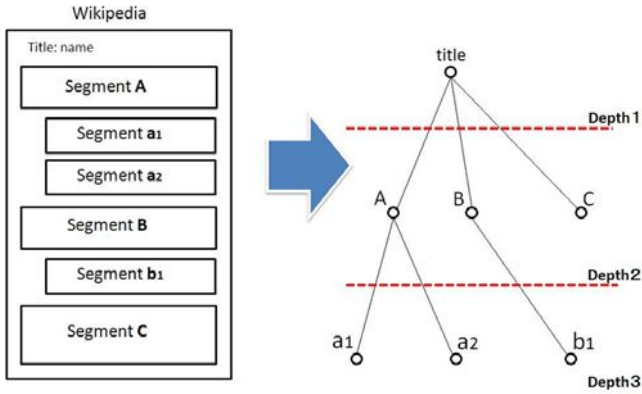
Fig. 1. Display of GSS

input topic extends over multiple articles in English Wikipedia, we extract target articles based on the Wikipedia link graph and our proposed relevance degree.

Link Structure Analysis

First, we create a link graph for English Wikipedia based on the user's input keyword as follows:

1. The system extracts English articles that have the same title as the user's input. We designate the English article as the basic article. We regard the basic article as the root node of the link graph.
2. The system extracts all interactive linked articles that are the subjects of link-out and link-in connections with the root node. Then it includes the articles as nodes and thereby creates the link graph.
3. The system calculates the relevance degree between the root node and respective articles in the link graph.
4. When the relevance degree is greater than a threshold α value, then the system regards the articles as relevant articles.

Calculating Relevance Degree

We proposed extraction of the relevance article using only cosine similarity between the root node and other nodes. However, the result obtained using the proposed method is not good. For example, in the case of "Cricket." the article of "Batting (cricket)" is closely related to that of "Cricket." However, their similarity degree is low. For that reason, the system is unable to target the article as a relevant article.

As described next, we specifically examine the position of the link anchor in the pager. In general, an important anchor appears in the upper area of the page and also appears in the section area more than in a subsection area. We use that structural feature and calculate the relevance degree between the root node and other node in the link graph as follows:

1. The system divides the basic article according to the structure of the table of contents of the basic article. We designate the divided parts as segments.
2. As a result, it creates segment tree for which the root node is the title name of the basic article; child nodes are segments (Fig. 1). In the segment tree, the child node of the root node is a section of the basic article; the grandchild node is a subsection of the parent node. The left side nodes show younger segment numbers in the width.
3. The system extracts nodes that have an anchor.
4. It calculates the relevance degree. At this time, our hypothesis is that important anchors related to the basic article appear many times in the basic article. They appear in upper areas of the basic article, and also appear in the section area more than subsection area. We infer that the relevance degree of the nodes of the low hierarchy and left side in the segment tree are higher than the nodes of the deep hierarchy and right side in the segment tree. Therefore, we use the following expression to calculate the degree of relevance.

$$W_{kl} = \left\{ af \cdot \cos(k,l) + \sum_{i=1}^{af} \left(\frac{1}{d_i} \right)^{n_i} \cdot (n_i - o_i + 1) \right\} / \max(W_{km}) \qquad (1)$$

In the equation shown above, af is the number of the anchor which is interactive link of the article l in a basic article k. $\cos(k,l)$ represents the cosine similarity between k and l. d_i is the deepness of node i in a segment tree, and n_i is a sibling node of i. o_i is a sequence number of i in the same hierarchy in the segment graph. $\max(W_{km})$ signifies the maximum value in all linked articles linked from the basic article.

When relevance degree W_{kl} is greater than the threshold α value, it becomes a comparison target article.

3.2   Comparison between Japanese Article and English Articles

Almost all Wikipedia articles are divided into segments based on the table of contents, meaning that the segments are divided semantically. When comparing the similarity of multilingual Wikipedia, we examine the segment of the table of contents of Wikipedia specifically. Particularly, we compare each article based on the each segment.

First, we extract nouns from respective articles. Next we translate English nouns to Japanese nouns using an English–Japanese dictionary. We use GENE95[2], Google Ajax API[3], Microsoft Translator API[4], and Wikipedia. As described in this paper, we are unconcerned about word sense disambiguation. That is left as

---

[2] GENE95 http://www.namazu.org/ tsuchiya/sdic/data/gene.html
[3] Google Ajax API http://code.google.com/apis/language/
[4] Microsoft Translator API http://www.microsofttranslator.com/dev/

Table 1. Results of experiment 1

Query	Baseline (Cosine measure)			Our proposed method		
	Prec. (%)	Rec. (%)	F	Prec. (%)	Rec. (%)	F
Cricket	100%	40	50	97	41	58
Warwick Castle	0	0	0	25	50	33
Snooker	100	7	13	70	47	56
Fish and chips	67	20	31	50	60	55
Goodwood Festival of Speed	0	0	0	100	100	100
Polo	0	0	0	100	75	86
Association football	100	33	50	100	78	86
Boxing Day	0	0	0	0	0	0
Flag of Scotland	100	17	29	100	33	50
Gaelic handball	0	0	0	67	100	80
Kipper	0	0	0	75	100	86
National Gallery of Scotland	0	0	0	59	100	74
Sport	100	8	0	100	67	80
Table tennis	0	0	0	100	83	91
Volleyball	22	40	29	44	44	44
Average	37	10	13	68	61	61

a subject for future work. Then we calculate the similarity between the Japanese article and each English article using cosine similarity.

If the cosine similarity is less than the threshold value, then it becomes difference information.

4   Experimental Evaluation

Our experiment confirmed the accuracy of difference information extraction methods using precision, recall, and F value by comparison of our proposed method with the baseline. The important points of our method are that we calculate the relevance degree using not only cosine similarity but also the number of each anchor and position of the anchor in a basic article. To assess the availability of our method, we use the baseline, which is only the cosine similarity between basic articles and other linked articles. We set the threshold α to 0.2 based on the result of our pre-experiment.

Experimental results are presented in Table 1. In this table, Prec. means precision ratio, Rec. means recall ratio, and F means F-measure. When we compare the results of baseline with our proposed method, we can observe that the average of precision ratio improves from 37% to 68%, and that the average of recall ratio improves from 10% to 61%. Moreover, the average of F-value improves from 13% to 61%. From these improvements, we confirmed that our proposed method can extract appropriate parts of articles from English Wikipedia articles.

We can explain what inappropriate articles are extracted by the baseline, but which are not extracted by our method. If the query is "National Gallery of

Scotland," then the section about the "Royal Scottish Academy" is appropriate as a relevant section because this section is about a museum. However, the section related to "Seven Sacraments (Poussin)" is inappropriate because this section describes religion, not a museum. Both sections have many terms that are used at the "National Gallery of Scotland." Therefore, the previous method determines that these two articles are relevant sections. In contrast, our proposed method uses the Wikipedia link structure. When we consider the section "Seven Sacraments (Poussin)," the terms which appear at both this section and in the "National Gallery of Scotland" appear at the bottom of the section. Therefore, the weights of these terms are low. This section is therefore regarded as not relevant by our proposed method, whereas this section was determined as relevant by the cosine-measure-based method. "Royal Scottish Academy" is determined as relevant by both the cosine-measure-based method and our proposed method. We observe these decisions in many cases. The accuracy of our proposed system is better than that of the cosine-measure-based method.

Moreover, when we use the queries "Association Football" and "Table Tennis," which are generally used terms, our proposed system is more effective than when we use terms for specific fields because, when we use general terms, the relevant terms is numerous. Also, the links to Wikipedia articles are numerous. Therefore, the relevance of correct sections is large.

5 Conclusion

In this paper, we proposed a method for extracting information which exists in one language version, but which does not exist in another language version. Our method specifically examines the link graph of Wikipedia and structure of an article of Wikipedia. Then we extract comparison target articles of Wikipedia using our proposed degree of relevance. We conducted an experiment to show the accuracy of our proposed relevance degree by comparing cosine similarity. Results show that our proposed relevance degree is superior to that obtained using existing methods.

Future work will include the following tasks.

– Calculating credibility of Wikipedia articles
  Our proposed method extracts and presents difference information between different language versions of Wikipedia. However, Wikipedia credibility is not always good. For that reason, we must assess Wikipedia credibility in future studies.
– Considering word sense disambiguation
  Many instances of word sense disambiguation exist, but we ignored such cases in this paper. We intend to consider word sense disambiguation in later investigations.

References

1. Adar, E., Skinner, M., Weld, D.S.: Information arbitrage across multi-lingual wikipedia. In: Proceedings of the Second ACM International Conference on Web Search and Data Mining, WSDM 2009, pp. 94–103. ACM, New York (2009)
2. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: WWW7: Proceedings of the Seventh International Conference on World Wide Web 7, WWW7, vol. 30, pp. 107–117. Elsevier Science Publishers B. V. (1998)
3. Chen, Z., Liu, S., Wenyin, L., Pu, G., Ma, W.Y.: Building a web thesaurus from web link structure. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, pp. 48–55 (2003)
4. Kamps, J., Koolen, M.: Is wikipedia link structure different? In: Proceedings of the Second ACM International Conference on Web Search and Data Mining, pp. 232–241 (2009)
5. Milne, D.: Computing semantic relatedness using wikipedia link structure. In: Proc. of New Zealand Computer Science Research Student Conference, NZCSRSC 2007. CDROM (2007)
6. Milne, D., Medelyan, O., Witten, I.H.: Mining Domain-Specific thesauri from wikipedia: A case study. In: WI 2006: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 442–448 (2006)
7. Nakatani, M., Jatowt, A., Tanaka, K.: Adaptive ranking of search results by considering user's comprehension. In: Proceedings of the 4th International Conference on Ubiquitous Information Management and Communication, ICUIMC 2010. CDROM (2010)
8. Nakayama, K., Hara, T., Nishio, S.: Wikipedia Mining for an Association Web Thesaurus Construction. In: Benatallah, B., Casati, F., Georgakopoulos, D., Bartolini, C., Sadiq, W., Godart, C. (eds.) WISE 2007. LNCS, vol. 4831, pp. 322–334. Springer, Heidelberg (2007)
9. Strube, M., Ponzetto, S.P.: WikiRelate! computing semantic relatedness using wikipedia. In: Proceedings of the 21st International Conference on Artificial Intelligence (AAAI 2006), pp. 1419–1424 (2006)
10. Takahashi, Y., Ohshima, H., Yamamoto, M., Iwasaki, H., Oyama, S., Tanaka, K.: Evaluating significance of historical entities based on tempo-spatial impacts analysis using wikipedia link structure. In: Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia, HT 2011, pp. 83–92. ACM, New York (2011)

Multi-strategic Approach of Fast Composition of Web Services[*]

Gang Wang, Li Zhang, and Kunming Nie

Department of Computer Science and Engineering, University of BUAA, Beijing, China
f_lag@cse.buaa.edu.cn, lily@buaa.edu.cn, nkm1985@qq.com

Abstract. There are several approaches for QoS–aware web service composition. But most approaches are concerned about web service composition algorithm itself, while ignoring the flexibility for user to set QoS and cost. Most of them require QoS constraints given in form of numbers. In reality, it is difficult for users because they don't know the exact value or range of QoS of composite web services. Users may just want composite solutions of web services in different degrees of QoS according to their economics or want composite solutions in quality priority or cost priority. To solve non-clarity and diversity of user's QoS requirements, this paper proposes a multi-strategic approach of fast composition of web services. With the approach, users can get solutions of web service composition quickly and as satisfied as possible. Or they become gradually clear about the range of QoS they want and finally find satisfied composite solutions.

Keywords: multi-strategic, web service composition, fuzzy, GA.

1   Introduction

SOC (Service-Oriented Computing) becomes a new computational model for developing information system in the open heterogeneous environment. SOA (Service-Oriented Architecture) solves the problem of SOC technical architecture by coupling various web services loosely and fast to adapt their business processes to meet new requirements. Web services as a basic technology for SOA have been more and more used. With various web services provided, selection of web services for composition becomes an important research topic.

There are several QoS-based approaches of web service composition. But most approaches are concerned about web service composition algorithm, while ignoring the flexibility for user to set QoS and cost. Most of them require QoS given in the form of numbers such as [1, 2, 3]. They are based on a hypothesis: users can explicitly give or describe their non-functional requirements (QoS and cost). But in fact, users may not know the exact value or range of QoS. Or they may not have the specific

---

requirement of QoS. They, like buying cars, just want the composition of web services in high QoS, medium high QoS, very high QoS and so on according to users' economics. To solve non-clarity and diversity of user's QoS requirements, we refer approaches in web service composition and in semantic QoS-based web service selection using fuzzy logic [4, 5], integrate them to propose a multi-strategic approach of fast composition of web services for recommendation for solutions to web service composition.

The remainder of this paper is organized as follows. Section 2, the main part of this paper details the multi-strategic approach of fast composition of web services. At the beginning of Section 2, it describes the multi-strategic approach on the whole. Then Section 2 is divided into 4 parts to describe every strategy. Finally, Section 3 concludes.

2   Multi-strategic Approach of Fast Composition of Web Services

Fig. 1 shows the process of using the multi-strategic approach of fast composition of web services. The multi-strategic approach of fast composition of web services has 4 strategies for web service composition. They are (1) hotness-based web service composition, (2) fuzzy NFR-based web service composition, (3) user-preference-based web service composition and (4) web service composition with clear-fuzzy mixed NFR (Non-Functional Requirement).
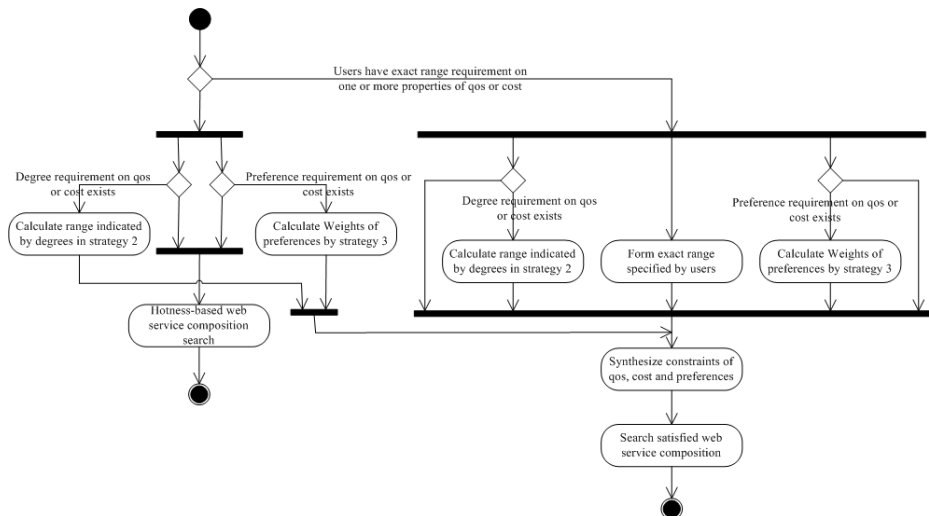
Fig. 1. Users can use one or more strategies to get solutions of web service composition according to clarity of users' requirement of QoS and cost of web services

To complete a complex task which may emerge due to change of business process or business environment, users need to composite web services. In this situation, users usually know how to divide the complex task into component tasks, and users can describe the combination of tasks by using some workflow description languages such as BPEL4WS (Business Process Execution Language for web services). But users may have no idea about constraints of QoS and cost. For this situation, this paper provides three strategies of fast web service composition.

If users have no requirement on QoS and cost, they can use Strategy 1 of the hotness-based web service composition to get solutions of web service composition. The hotness-based web service composition is a web service composition strategy based on VI (Visits Index), which can indicate the intensity of visits of web services and thus indicate popularity or hotness of web services. The hotness-based web service composition is explained in Section 2.1.

If users have priorities of QoS and cost, they can use Strategy 3 of the user-preference-based web service composition to get solutions of web service composition by setting preferences of properties of QoS and cost. Strategy 3 is explained in Section 2.3.

If users have not clear requirement on QoS and cost which means users' requirement cannot be expressed in exact numbers, and they just want some properties of QoS and cost to reach the degrees such as high, very high and so on among all possible solutions, they can use Strategy 2 of the fuzzy NFR-based web service composition to get solutions of web services by setting degrees of properties of QoS and cost. More details of the fuzzy NFR-based web service composition are explained in Section 2.2.

If Strategy 1, 2 and 3 cannot meet users' requirements of QoS and cost, users can use Strategy 4 of the web service composition with clear-fuzzy mixed NFR to get solutions of web services. Strategy 4 allows users to set the QoS and cost constraints using exact range of QoS. This situation may appear when users encounter QoS problems in the process of using the composite web services produced by Strategy 1, 2 or 3 and they want to get a new web service composition to meet the requirement of QoS and cost. Strategy 2, 3 and 4 can be used simultaneously. Strategy 4 is explained in Section 2.4.

The choice of services is an iterative process and different strategies can be mixed. With the four strategies, users can get solutions of web service composition fast and as satisfied as possible or they can gradually make the requirement of QoS and cost of web services more and more clear in the processing of using the composite web services produced by the four strategies and finally get the solutions of web service composition which meet their requirement of QoS and cost.

2.1 Fast Web Service Composition Strategy 1: Hotness-Based Web Service Composition

Scenario: A user wants to fulfill a complex task with web services, but he doesn't know the exact value or range of QoS that he wanted or he has not the non-functional requirement. He just wants a composite solution of web services which are used by most people.

This paper proposes hotness-based web service composition. The hotness or popularity of web services indicates the intensity of visits of web services. The more hot or popular a web service is, the more people it can meet in requirement. It should be recommended first. We use VI (Visits Index), to indicate the hotness or popularity.

In our previous work [11], we introduce VI to indicate popularity or hotness of use of web services. It is calculated by the following formula:

$$VI(s) = (\sum_{i=1}^{n} \omega_i * x_i) / \sum_{i=1}^{n} t_i,$$

$$\sum_{i=1}^{n} \omega_i = 1 \text{ and } 0 \le \omega_i \le 1 \text{ and } \forall i \le j, \omega_i \le \omega_j \qquad (1)$$

where VI is the visits index, s is a web service, n is the number of sampling, x_i is visits of web services in the period of t_i, ω_i is weight assigned to x_i. The requirement $\forall i \le j, \omega_i \le \omega_j$ expresses visits in period t_j indicates the trend of visits of web services more recently than t_j. Give more weight to more recent period can make VI have the ability to indicate the rate of visits increment or decrease of web services. Every t_j may be not equal while considering intensity of visits. The determination of n and length of every period t depends on specific condition.

Algorithm of Strategy 1: Assume a complex task T with n tasks, represented as T={ t_1, t_2,…, t_n }, and each task has candidate web services, represented as t_i={s(i,1),s(i,2),…,and s(i, n_i)}. For each task, we select the web service with highest VI from the candidate set of web service of the task. The selected web services are the composite solution recommended to users.

Although the composite solution, named as the composite web service may not be absolutely the best solution, it has high possibility to become the hottest or most popular web service, which has good characteristics of QoS and cost attracting people to use, because every component web service has the highest VI among the candidate web services of the corresponding task.

2.2   Fast Web Service Composition Strategy 2: Fuzzy NFR-Based Web Service Composition

Scenario: To complete a complex task, users want to composite web services. But users still don't know the exact value or the range that they want. They just want web services with high QoS, medium high QoS, very high QoS and so on according to users' economics.

This paper proposes fuzzy NFR-based web service composition. In the approach based on fuzzy information, each property of QoS and cost of web services are divided into several parts. Each part of a property indicates a degree of the corresponding property. Users select one or more degrees of the properties, and then the solutions of web service composition come out.

Fuzzy Division of QoS and Cost of Web Services
Referring to the conception of interval number in fuzzy mathematics, this paper divides the values of QoS and cost of web services into several subsets to represent classes of QoS and cost of web services.

For a property, labeled as P, the property is divided into N parts of $\{x_1, x_2, \ldots, x_i, \ldots, x_N\}$. The number N is decided according to the actual situation. N should be not too big, which means the property should be not divided into many parts because that confuses users' selection and betrays the principle of fast composition. Each part represents the degree of the property.

Assuming the range of x_i is $[l_i, h_i]$, let $m_i=(l_i+h_i)/2$.

We have the following requirement: for any x_i, x_j, if i>j, then $l_i >= m_j$ and $m_i >= h_j$.

The requirement not only reflects vague border of different degrees, but also reflects certainty or distinctness between degrees.

For example, we assume the range of RT is from 40ms to 160ms. Then RT (response time), a property of QoS, could be divided into 5 parts. Five parts represent the degrees of very low RT, low RT, medium RT, high RT and very high RT. The borders of five parts are in Table 1.

Table 1. Degrees and ranges of RT

Response Time	very high (VH)		high(H)		medium(M)		low(L)		very low (VL)
interval	[40,70]		[65,95]		[85,115]		[105,135]		[130,160]
overlap		5		10		10		5	
Non- overlap	25		15		10		15		25

And we adopt the following principles and assumptions to divide the ranges: (1) N, the number of parts, is not bigger than 10; (2) The higher or the lower the degree is, the more values that are not overlapped by other degrees the degree should have. This means VH or VL are easy to identify while H, M and L are not; (3) The higher or the lower the degree is, the more distinct the border between degrees is.

When users select one or more degrees for each properties of web service, it determined constrained condition. Then constraints are calculated as follows:

$$p^u \in \bigcup \text{Range}(d_v^u) \tag{2}$$

where p^u is a non-functional property of web services and u is the name of the property such as RT, cost, d_v^u is the degree of p^u and v is the name of the degree such as "high" or "very high", and Range is a function which turns the degree description of a property into the corresponding numeric range of the property.

Using GA to search for optimization solutions of web service composition
After the users select the degrees of QoS and cost of web services, the process of web service composition begins. With constraints of users' QoS and cost, selection from the candidate set of web service of each subtask is constraint satisfaction problem. There are several algorithms for these problems. This paper adopts genetic algorithm (GA), which can solve global optimization problems rapidly.

The optimization objective is to find optimal solutions of web service composition which has the best trade-off, known as a Pareto set. It indicates best balance between high quality and low cost within the constraints users set.

Genome coding. To use the GA to search for a solution of the web service composition, we first need to design a genome in GA to represent the problem. Each gene g_i represents a subtask t_i. And the g_i takes the integral value from 1 to n_i the number of candidate web services for the subtask t_i. We label candidate web services for a subtask from 1 to n_i. If the value of the gene is j, $0<j<=n_i$, it means selection of web service j to complete the subtask t_i.

Fitness evaluation. To measure individual the degree near optimal solution, we need to define a fitness function. We choose the Cobb-Douglas function [10] as fitness function, which evaluates the composite solutions is the best solutions.

Formula 3 is the function prototype.

$$u(x,y) = x^a y^b, a > 0, b > 0 \tag{3}$$

Cobb-Douglas function is one of the most widely used utility function, with its special feature which reflects users' preference have the following characteristics: First, decision maker think that the more value of the property is, the better the solution is, some properties which is negative correlative with the utility of u such as cost we can use methods to make it positive correlative, secondly, for a specific value u, it is better for all properties contribution to u as equal as possible than one or several properties contribution to u far more than others. Because these features reflect characteristics of thinking when making decisions in our daily lives, this paper uses the Cobb-Douglas utility function as decision-making function.

Before using Cobb-Douglas function, data should be preprocessed. Different properties of QoS and cost of web services have different dimension. In our present system, we consider three QoS (response time, availability, reliability) and cost. To remove the effect of different dimensions, data should be normalized to make different properties into the same extent of 0 to 1.

For the positive correlative properties, this paper uses the following normalization function

$$np_{i,j} = (p_i - P_i^{min})/(P_i^{max} - P_i^{min}) \tag{4}$$

For the positive correlative properties, this paper uses the following normalization function

$$np_{i,j} = (P_i^{max} - p_i)/(P_i^{max} - P_i^{min}) \tag{5}$$

where p represents a property of QoS and cost of web services, $p_{i,j}$ is the value of   a property of QoS of the candidate web service s(i, j) of the subtask t_i, p_i^{max} is the maximum of the property p in the candidate web service set of the subtask t_i and p_i^{min} is the minimum. Then P_i^{max} is p_i^{max} plus several percentage points of (p_i^{max}- p_i^{min}), P_i^{min} is p_i^{min} minus several percentage points of (p_i^{max} - p_i^{min}) so that normalized properties are **not zero** and can be used in Cobb-Douglas function. In this paper, we take 10 percentage points. It may be adjusted according to actual situation.

Then Cobb-Douglas function used as fitness function is:

$$u = \text{nrt}^{\omega_1}\text{na}^{\omega_2}\text{nr}^{\omega_3}\text{nc}^{\omega_4}, \sum_{i=1}^{4}\omega_i = 1 \text{ and } 0 \le \omega_i \le 1 \tag{6}$$

where nrt, na, nr and nc represent normalized response time, normalized availability, normalized reliability and normalized cost of composite web services. ω_1, ω_2, ω_3, and ω_4 indicates preference weights of response time, availability, reliability and cost, which are specified by the system or users.

Response time, availability, reliability and cost of composite web services in the Formula 6 are calculated based on QoS of component web services as follows:

Table 2. QoS Calculation of composite web services

	sequence	parallel	switch	loop
response time	$\sum_{i=1}^{m} \text{rt}_i$	$\text{Max}_{i=1}^{p}(\text{rt}_i)$	$\sum_{i=1}^{n}(p_i * \text{rt}_i)$	$k*\text{rt}$
availability	$\prod_{i=1}^{m} a_i$	$\prod_{i=1}^{p} a_i$	$\sum_{i=1}^{n}(p_i * a_i)$	a^k
reliability	$\prod_{i=1}^{m} r_i$	$\prod_{i=1}^{p} r_i$	$\sum_{i=1}^{n}(p_i * r_i)$	r^k
cost	$\sum_{i=1}^{m} c_i$	$\sum_{i=1}^{p} c_i$	$\sum_{i=1}^{n}(p_i * c_i)$	$k*c$

In Table 2, m represents the number of tasks in a sequence construct, p represents the number of parallel tasks in a parallel construct, p_i represents the probability of the case i in a switch construct where $\sum_{i=1}^{n} p_i = 1$, k represents the estimated number of iterations in a loop construct. The calculation method of QoS and cost of composite web services is adopted from [7] and [8].

The selection function is the most used roulette selection. And the crossover function is the standard two-points crossover [9] while the mutation function randomly selects a subtask t_i (i.e., a gene in the genome) and then randomly select another web service to replace from t_i.

2.3   Fast Web Service Composition Strategy 3: User-Preference-Based Web Service Composition

Scenario: Users may have different preferences with regard to cost and quality to web services. Or users may pay more attention of one or more properties such as response time than other properties.

This approach supports this by giving different weights to different properties of web services. The more the weight of a property is, the more important the property is. In the fast web service composition Strategy 3, we still use the GA algorithm described in the fast web service composition Strategy 2 by setting the weights in fitness function to fulfill the preferences. There are several approaches to determine weights. Here we choose AHP (Analytic Hierarchy Process) [12,13] because it makes quantitative analysis of qualitative issues based on pair-wise comparison other than direct comparison and it is simple, flexible, practical and widely used method of multiple criteria decision making.

In the Strategy 3, we still consider three QoS (response time, availability, reliability) and cost.

And the fitness function is

$$u = nrt^{\omega_1} na^{\omega_2} nr^{\omega_3} nc^{\omega_4}, \sum_{i=1}^{4} \omega_i = 1 \text{ and } 0 \leq \omega_i \leq 1 \tag{7}$$

In our present system, we give two ways for users to determine the priorities. One is the simple way. Users can select cost priority or quality priority. And the system provide three degrees for priorities that is I (important), VI (very important), and VII (very very important). The three degrees represent corresponding relative weights of 2, 3 and 5. We use AHP to turn semantic descriptions into numeric weights. If users set preferences in the simple way, the system assume the QoS properties of response time, availability and reliability have the same priority.

If users select quality priority, then the weights is calculated in Table 3. Each row represents the corresponding weights of the degrees.

Table 3. Weights in the case of quality priority

	response time	availability	reliability	cost
I	0.2619	0.2619	0.2619	0.2143
VI	0.2725	0.2725	0.2725	0.1825
VVI	0.2899	0.2899	0.2899	0.1303

If users select cost priority, then the weights is calculated in Table 4.

Table 4. Weights in the case of cost priority

	response time	availability	reliability	cost
I	0.2369	0.2369	0.2369	0.2893
VI	0.2226	0.2226	0.2226	0.3322
VVI	0.1914	0.1914	0.1914	0.4258

If the simple and fast way doesn't meet users' requirement, the other way can provide complex priority determination. Users can sort QoS and cost of web services according to users' preferences. User can set priorities of two properties equal or one higher than the other. If users set the priority of one property higher than that of the other, users can set degrees of priority further with three classes of I(important), VI(very important), and VII(very very important).

The properties that users don't set priority will be set the lowest priority on the assumption that users have no requirement on these properties.

2.4   Fast Web Service Composition Strategy 4: Web Service Composition with Clear-Fuzzy Mixed NFR

The above 3 strategies may still not meet users' requirement. The situation exists when users find the exact range of QoS and cost of web services after they experience the composite web services produced by the above strategies or when users know the exact range of QoS and cost at first. Users can use Strategy 4 to set the exact numerical range that they want.

In Strategy 4, users can still set users' preferences of QoS and cost web services like the way in Strategy 3, and constrain the properties of QoS and cost of web services using fuzzy degree selection in Strategy 2. The relations between Strategy 2, 3 and 4 are shown in Fig. 1 in the beginning of Section 2.

3   Conclusion

This paper proposes multi-strategic approach of fast composition of web services for recommendation for solutions to web service composition. And the usage and relations of the four strategies is shown in Fig. 1. In the process of using the four strategies, users either get their satisfied web service composition rapidly, or know their exact requirement of QoS and cost of composite web service iteratively and finally get their required web service composition.

References

1. Canfora, G., Penta, M.D., Esposito, R.: An approach for QoS-aware service composition based on genetic algorithms. In: Proceeding of the 2005 Conference on Genetic and Evolutionary Computation, pp. 1069–1075. ACM Press, New York (2005)
2. Zhang, L.-J., Li, B.: Requirements driven dynamic services composition for Web services and grid solutions. Journal of Grid Computing 2, 121–140 (2004)
3. Xia, H., Li, Z.-Z.: A Particle Swarm Optimization Algorithm for Service Selection Problem Based on Quality of Service in Web Services Composition. Journal of Beijing University of Posts and Telecommunications 32, 63–67 (2009)
4. Wang, H.-C., Lee, C.-S., Ho, T.-H.: Combining Subjective and Objective QoS Factors for Personalized Web Service Selection. Expert Systems with Applications 32, 571–584 (2007)
5. Wu, Z.-P., Yuan, M.: User-Preference-Based Service Selection Using Fuzzy Logic. In: International Conference on Network and Service Management, pp. 342–345. IEEE Press, Niagara Falls (2010)
6. QoS for Web Services: Requirements and Possible Approaches, http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/
7. Cardoso, J.: Quality of Service and Semantic Composition of Workflows. PhD thesis, Univ. of Georgia (2002)
8. Zeng, L.-Z., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-aware middleware for web services composition. IEEE Transactions on Software Engineering 30, 311–327 (2004)
9. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Pub. Co. (1989)
10. Wang, Z.-K., Ou, R.-Q.: Microeconomics Diagram, pp. 32–58. China Renmin University Press (2005)
11. Wang, G., Nie, K.: A Framework of VI-based Ranking and Recommendation of Web Services. In: 6th International Conference for Internet Technology and Secured Transactions, Abu Dhabi, UAE (2011) (in press)
12. Zuo, M., Wang, S., Wu, B.: Research on web services selection model based on AHP. In: IEEE International Conference on Service Operations and Logistics, and Informatics, pp. 2763–2768. Inst. of Elec. and Elec. Eng. Computer Society, Beijing (2008)
13. Saaty, T.L.: A scaling method for priorities in hierarchical structures. Journal of Mathematical Psychology (1977)

Collaborative Filtering via Temporal Euclidean Embedding

Li'ang Yin, Yongqiang Wang, and Yong Yu

Computer Science Department, Shanghai Jiao Tong University
Dongchuan Road, Minhang District, Shanghai, China
{yinla,wangyq,yyu}@apex.sjtu.edu.cn

Abstract. Recommender systems are considered as a promising approach to solve the problem of information overload. In collaborative filtering recommender systems, one of the most accurate and scalable algorithms is matrix factorization. As an alternative to this popular latent factor model, Euclidean embedding model presents the relationship between users and items intuitively, and generates recommendations fast. In this paper, a temporal Euclidean embedding (TEE) model is proposed by incorporating temporal factors of rating behavior. Through experiments on Netflix and Movielens data sets, we show the improvement of prediction accuracy, while keeping the efficiency of recommendation generation.

Keywords: Collaborative filtering, temporal Euclidean embedding, fast recommendation generation.

1 Introduction

Living in an era with exploding amounts of information, people have strong needs to find their way through the digital ocean. The goal of a recommender system is to automatically recommend appropriate items for a specific user. Collaborative filtering (CF) systems make recommendation based on the rating behavior of users [1]. With no need of explicit user profiles or item contents, CF has capacity of uncovering complex and unexpected patterns of user behavior. As a result, CF has attached much of the attention over the past decade [10].

Among model-based collaborative filtering methods, one of the most accurate and scalable algorithms is matrix factorization (MF). MF assumes users and items can be embedded in a latent space, and uses dot product of user and item latent vectors to predict ratings. As an alternative to MF, Khoshneshin et al. proposed a novel approach called Euclidean embedding (EE), which embeds users and items in a unified Euclidean space [7]. In Euclidean space, the distance between a user and an item is inversely proportional to the rating. Thus it is intuitively understandable that a user may like the closest items. Moreover, KNN search methods can be adopted to generate recommendations fast.

Euclidean embedding model considers user rating as static behavior, however, real-world data sets like Netflix[1] and Movielens[2] show that user rating behavior is time-dependent [12]. Give an example of user rating changes over time on a movie recommender system: a user rated 3 stars as his average rating level; after one year, due to the improvement of recommender system, he received more appropriate movies; therefore this user rose the average to 3 and a half stars. Items such as presents tend to get higher ratings during holidays; movies are rated higher as they become classic. These days, books about Steve Jobs are more popular than usual. The effect of temporal factors is shown too important to be ignored, as a result, recent research works of recommender systems usually take temporal information into account. In order to achieve a more accurate recommender system while keeping the advantages of Euclidean embedding, we propose a temporal Euclidean embedding (TEE) model by incorporating temporal factors.

The rest of the paper is organized as follows: In the next section related literatures are discussed. In Section 3, we formally describe matrix factorization and Euclidean embedding methods. We propose the temporal Euclidean embedding model in Section 4. Experiments and results are presented in section 5. Last section is the conclusion of this paper.

2   Related Work

There are two general classes in collaborative filtering recommender systems: neighborhood and model-based [2]. Singular value decomposition (SVD) [3,4] or matrix factorization [5] has gained popularity among model-based recommender systems since the Netflix Prize competition [11] and is one of most accurate and scalable collaborative filtering algorithms.

Recently, an alternative named Euclidean embedding is proposed to intuitively measure the relationship between users and items and fast generate recommendations [7]. In the data mining and machine learning area, Euclidean embedding has been frequently used as a visualization approach [8], a dimensionality reduction technique [9] and a unsupervised learning method [6].

Recent literatures have noticed the temporal effect of recommender systems. Ding and Li suggest a time weighting scheme for a similarity-based collaborative filtering system [14]. The ratings of previous items are decayed as time difference increases. Koren proposes a variety of temporal factors, and incorporates them into the matrix factorization approach to achieve a precise predictive model [12]. Lu et al. and Xiong et al. consider time as one additional dimension [15,16]. In their works, matrix and tensor factorization are adopted in modelling respectively. Lathia et al. take a deep analysis into the temporal diversity of recommender systems [13].

To the best of our knowledge, temporal factors have not been incorporated in Euclidean embedding framework. This constitutes the motivation of our work.

---

[1] http://www.netflixprize.com
[2] http://www.grouplens.org

3   Euclidean Embedding

3.1   Matrix Factorization

In a collaborative filtering problem, there are N users and M items. Each user has rated some of the items. Assume one item is rated only once by one user, we denote r_{ui} the rating of user u for item i. To predict unknown ratings, MF learns a model that minimizes the root mean squared error (RMSE) between predictive and known ratings:

$$\min \sum_{u,i} w_{ui}(r_{ui} - \hat{r}_{ui})^2 \tag{1}$$

where $\hat{r}_{ui}$ is the predictive rating of user u for item i. w_{ui} is 1 if rating r_{ui} is known and 0 otherwise.

Matrix factorization associates each user u and each item i with a user-factor vector $p_u \in \mathbb{R}^f$ and an item-factor vector $q_i \in \mathbb{R}^f$ respectively. The prediction of $\hat{r}_{ui}$ is the dot product of user and item-factor vectors:

$$\hat{r}_{ui} = \bar{\mu} + b_u + b_i + p_u^T q_i \tag{2}$$

$\bar{\mu}$ is the average of all known ratings. b_u and b_i are the biases of user u and item i respectively.
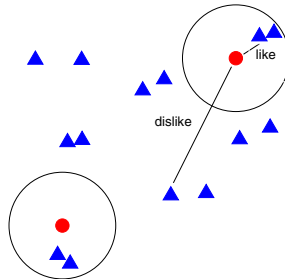
Fig. 1. An illustration of Euclidean embedding. Users are red circles and items are blue triangles. A big circle surrounding the user indicates the candidate area. Choosing items from the candidate area speeds up the recommendation.

3.2   Euclidean Embedding

As an alternative to MF approach, we assume that users and items are embedded in a unified Euclidean space. Each user u and each item i are presented by their coordinates x_u and y_i respectively, therefore the similarity or preference between users and items is intuitively understandable (see Fig. 1). By Euclidean embedding, the dot product in (2) is replaced with minus squared Euclidean distance of x_u and y_i:

$$\hat{r}_{ui} = \bar{\mu} + b_u + b_i - (x_u - y_i)^T (x_u - y_i) \tag{3}$$

In the Euclidean space, since items with high ratings are closest to the user, a local search is sufficient to retrieve top-k desirable items. This property leads to fast recommendation generation.

4   Temporal Euclidean Embedding

For the sake of achieving a more accurate recommender system, we propose a temporal Euclidean embedding model by incorporating temporal information. To do this, we first introduce temporal factors into matrix factorization to yield a temporal matrix factorization (TMF) model, which is also compared in our experiments. Further we embed factor vectors in Euclidean space to derive a TEE model.

4.1   Temporal Matrix Factorization

By introducing the timestamp of ratings, a time-dependent rating is denoted as r_{uit} given by user u for item i at time t. We mainly take two temporal factors into account: temporal user bias and temporal item bias.

Temporal user bias b_{ut} can be considered as combination of a static user bias and a time-aware offset which presents the changes of user preference over time:

$$b_{ut} = b_u + c_u^T d_t$$

where b_u is the static user bias, c_u and d_t are factor vectors of user u and time t respectively.

The temporal item bias can be written in the similar manner as follows:

$$b_{it} = b_i + m_i^T n_t$$

where b_i is the static item bias. m_i and n_t are factor vectors of item i and time t respectively.

Therefore a predictive formula of TMF is:

$$\hat{r}_{uit} = \bar{\mu} + b_u + b_i + c_u^T d_t + m_i^T n_t + p_u^T q_i \tag{4}$$

4.2   Temporal Euclidean Embedding

Similar to the embedding of user and item factors in (3), temporal factors can be embedded in the Euclidean space as well. Consider factor vectors in (4) as coordinates in Euclidean space, we rewrite dot products in temporal bias in the manner of squared Euclidean distance respectively:

$$d_{ut} = (c_u - d_t)^T (c_u - d_t)$$
$$d_{it} = (m_i - n_t)^T (m_i - n_t)$$

The predictive formula of TEE by incorporating temporal user and item biases is derived as follows:

$$\hat{r}_{uit} = \bar{\mu} + b_u + b_i - d_{ut} - d_{it} - d_{ui} \tag{5}$$

We write $d_{ui} = (x_u - y_i)^T (x_u - y_i)$ to keep the formula simple and consistent. As in Euclidean embedding (3), the minus form of squared distance is adopted.

To avoid overfitting in training process, a regularizing term should be introduced in minimization problem (1), which yields:

$$\min \sum_{ui} w_{ui}\big((r_{uit} - \hat{r}_{uit})^2 + \lambda R\big) \tag{6}$$

where λ controls the strength of penalty and the regularizing term R is the magnitudes of parameters:

$$R = b_u^2 + b_i^2 + \|c_u\|^2 + \|d_t\|^2 + \|m_i\|^2 + \|n_t\|^2 + \|x_u\|^2 + \|y_i\|^2$$

We adopt gradient descent updating to learn model parameters. The updating rules for each parameter in (6) can be derived as follows:

$$b_u \leftarrow b_u + \eta(e_{uit} - \lambda b_u)$$
$$b_i \leftarrow b_i + \eta(e_{uit} - \lambda b_i)$$
$$c_u \leftarrow c_u - \eta\big(2e_{uit}(c_u - d_t) + \lambda c_u\big)$$
$$d_t \leftarrow d_t + \eta\big(2e_{uit}(c_u - d_t) - \lambda d_t\big)$$
$$m_i \leftarrow m_i - \eta\big(2e_{uit}(m_i - n_t) + \lambda m_i\big)$$
$$n_t \leftarrow n_t + \eta\big(2e_{uit}(m_i - n_t) - \lambda n_t\big)$$
$$x_u \leftarrow x_u - \eta\big(2e_{uit}(x_u - y_i) + \lambda x_u\big)$$
$$y_i \leftarrow y_i + \eta\big(2e_{uit}(x_u - y_i) - \lambda y_i\big)$$

where η is learning rate, $e_{uit} = r_{uit} - \hat{r}_{uit}$ is the error between given rating and predictive rating.

4.3   Fast Recommendation Generation

One of the advantages of temporal Euclidean embedding is fast generation of recommendations. In Euclidean space, the similarity or preference of a user and an item can be measured by the distance between them. In order to recommend k items to a user, first a KNN search is applied to find M candidates, where $M > k$. Then the system predicts ratings for all the candidates and retrieves top-k as final recommendations.

In TEE, we have three kinds of pair distances: user-item d_{ui}, user-time d_{ut} and item-time d_{it}. Since a system knows the time of recommendation, the distance of user-time is determined for a specific user. Therefore we need to choose candidate items by measuring d_{ui} and d_{it}.

Two approaches called fast-TEE and fast-plus-TEE are proposed based on different search standards. In KNN search process, fast-TEE only measures the distance of user-item and leaves calculation of d_{it} in prediction step. In contrast, fast-plus-TEE calculates the sum of d_{ui} and d_{it} to find candidates. The distances calculated in search step are used again in prediction to avoid repeating computing.

5   Experimental Results

We evaluate and compare four approaches in our experiments: MF, EE, TMF and
TEE. Experiments are taken on two popular data sets: Netflix and Movielens,
and implemented via Matlab on a dual core 2.53 GHz PC with 4 GB RAM.

5.1   Accuracy Comparison

The Netflix dataset consists of 100 million training ratings of over 480,000 users
for 17,770 movies. The ratings are collected from 1999 to 2005. To compare the
accuracy of models, we measure the RMSE on the probe. The Movielens is also
a movie rating dataset. We use the 1-million version which consists of 1 million
ratings for 3900 movies given by 6040 users. Rating data are collected from April
2000 to February 2003. We randomly select 20% data as test set.

We consider each day as a vector of temporal factors. Model parameters are
selected through 5-fold cross validation. We measure RMSE for all four methods
and experimental results by selecting different factor dimensionalities are shown
in Table 1. Two temporal methods outperform static ones respectively and the
prediction accuracy of TMF and TEE are similar. It's clear that the improvement
is due to incorporating temporal information.

Table 1. RMSE comparison on Netflix and Movielens dataset

Dataset	f	MF	EE	TMF	TEE
	10	0.9204	0.9220	0.9167	0.9163
Netflix	20	0.9146	0.9156	0.9036	0.9045
	50	0.9101	0.9116	0.9042	0.9037
	10	0.8541	0.8558	0.8519	0.8510
Movielens	20	0.8536	0.8553	0.8493	0.8484
	50	0.8518	0.8548	0.8484	0.8480
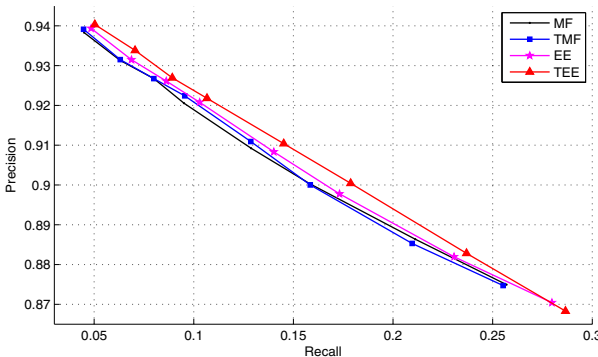
Fig. 2. Recall-precision curve of four methods, measured from $k = 20$ to $k = 200$

Besides RMSE, another set of measurements that are popular in retrieval problems are precision and recall. Follow the settings of [7], ratings of 4 and 5 are considered as desirable. The dimension of all methods are set to 50. We retrieve top-k recommendations and measure the precision and recall for value k. As k increases, recall increases while precision decreases. Figure 2 illustrates the precision and recall curve for different approaches on Movielens dataset. In precision and recall measurement, Euclidean embedding methods show an advantage over the matrix factorization ones. Between EE and TEE, the temporal one is better.

5.2   Fast Recommendation Generation

For a recommender system, the efficiency of finding recommendations for a user is also considerable. To show the speedup of Euclidean embedding, we compare the execution time of four methods: TMF, TEE, fast-TEE and fast-plus-TEE. The recommendation process for TMF and TEE is as follows: given a user u, we predict ratings for all items, and then recommend top-k items of highest ratings. Search strategies of fast-TEE and fast-plus-TEE have been described in Section 4.3.

Experiments are taken under settings of $k = 20$ and different M on Movielens dataset. Table 2 gives the experimental results, chosen of M is indicated after the approach name. Total recommendation time for all users is measured.

In general, fast-TEE is faster than fast-plus-TEE while fast-plus-TEE achieves higher precision and recall. When we set number of candidates to 500, fast-plus-TEE reduces the execution time by 87% almost without loss of precision and recall compared to TMF. The fewer the candidates, the faster the recommendation. As the number of candidates reduces to 100, fast-TEE spends only 1/30 execution time of TMF and remains high precision.

Table 2. Results of efficiency of recommendation generation

	Precision	Recall	Time (Sec)
TMF	0.9392	0.0447	136.4
TEE	0.9404	0.0503	119.8
fast+500	0.9348	0.0434	17.67
fast500	0.9326	0.0411	14.89
fast+200	0.9316	0.0236	9.617
fast200	0.9319	0.0229	6.759
fast+100	0.9276	0.0210	7.143
fast100	0.9251	0.0203	4.205

6   Conclusion

In this paper, we focus on understanding and improving Euclidean embedding model in collaborative filtering recommender systems. A temporal Euclidean embedding model is proposed by incorporating temporal user and item biases. With

the improvement of prediction accuracy, TEE benefits from fast recommendation generation as well. Fast recommendation adopts KNN search to retrieve candidates and further makes recommendation based on these candidates. The experimental results show that fast strategy reduces the recommendation time drastically.

References

1. Candillier, L., Meyer, F., Boulle, M.: Comparing State-of-the-Art Collaborative Filtering Systems. In: Perner, P. (ed.) MLDM 2007. LNCS (LNAI), vol. 4571, pp. 548–562. Springer, Heidelberg (2007)
2. Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In: Handbook on Recommender Systems. Springer, Heidelberg (2009)
3. Kurucz, M., Benczúr, A., Csalogány, K.: Methods for large scale SVD with missing values. In: KDD 2007: Netflix Competition Workshop (2007)
4. Peterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: KDD 2007: Netflix Competition Workshop (2007)
5. Takacs, G., Pilaszy, I., Nemeth, B., Tikk, D.: Matrix factorization and neighbor based algorithms for the Netflix Prize problem. In: Proc., 2008 ACM Conference on Recommender Systems (RECSYS 2008), pp. 267–274 (2008)
6. Borg, I., Groenen, P.: Modern multidimensional scaling. Springer, New York (1997)
7. Khoshneshin, M., Nick Street, W.: Collaborative filtering via Euclidean embedding. In: Recsys 2010 (2010)
8. Cox, M.A.A., Cox, T.F.: Multidimensional scaling. In: Handbook of Data Visualization. Springer Handbooks of Computational Statistics, vol. III, pp. 315–347 (2008)
9. Fodor, I.K.: A survey of dimension reduction techniques. Technical Report UCRL-ID-148494, Lawrence Livermore National Laboratory (2002)
10. Fisher, D., Hildrum, K., Hong, J., Newman, M., THomas, M., Vuduc, R.: Swami.: A framework for collaborative filtering algorithm development and evaluation. In: SIGIR 2000, Citeseer (2000)
11. Bennet, J., Lanning, S.: The Netflix Prize. In: KDD Cup and Workshop (2007), www.netflixprize.com
12. Koren, Y.: Collaborative filtering with temporal dynamics. In: KDD 2009 (2009)
13. Lathia, N., Hailes, S., Capra, L., Amatriain, X.: Temporal diversity in recommender systems. In: SIGIR 2010 Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (2010)
14. Ding, Y., Li, X.: Time weight collaborative filtering. In: Proc. 14th ACM International Conference on Information and Knowledge Management (CIKM 2004), pp. 485–492 (2004)
15. Lu, Z., Agalwal, D., Dhillon, I.S.: A spatio-temporal approach to collaborative filtering. In: Proceedings of the 3rd ACM Conference on Recommender Systems, RecSys 2009, pp. 13–20. ACM, New York (2009)
16. Xiong, L., Chen, X., Huang, T.-K., Schneider, J., Carbonell, J.G.: Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In: Proceedings of SIAM Data Mining 2010 (2010)

Transfer Learning with Local Smoothness Regularizer

Jiaming Hong, Bingchao Chen, and Jian Yin

School of Information Science and Technology, Sun Yat-Sen University,
Guangzhou, China
hongjiaming8888@gmail.com, superchan2006@163.com,
issjyin@mail.sysu.edu.cn

Abstract. The main goal of transfer learning is to reuse related domain data to learn models for the target domain. In existing instance-transfer learning algorithms, the relevance of instances is estimated mainly according to small amount of labeled instances, and the generalization ability of these algorithms needs to be improved. To make the relevance estimation more reliable, we propose to use unlabeled target domain instances as additional training data. These instances would serve as new domain knowledge sources to help determining the relevance of related domain instances. Under the universal framework of boosting, we introduce local smoothness regularizer, and obtain new empirical loss function, where unlabeled instances are included. Gradient decent method is used to iteratively optimize the loss function, and we finally obtain a new instance-transfer learning algorithm. Experiment results on text datasets show that the new algorithm outperforms competitive algorithms.

Keywords: Cross-Domain Learning, Transfer Learning, Boosting, Semi-Supervised Learning.

1 Introduction and Related Work

Transfer learning aims to address learning tasks in a new domain (called target domain), with the help of instances from another related domain (called source domain) [1,2]. The basic idea of instance-transfer learning is to re-weight the source domain instances according to their relevance to the target domain, and use them directly as new training instances [2,3]. In existing instance-transfer learning algorithms, since the relevance of each source domain instance is empirically estimated with the small amount of labeled target data, it might not generalize well on unseen test instances [2]. To address this problem, we propose to use unlabeled instances to help justifying the re-weighting process.

To incorporate domain knowledge from unlabeled instances, we use the similar strategy of semi-supervised learning [4,5]. Specifically, we use both prediction accuracy and local smoothness loss to evaluate whether a specific instance from source domain is relevant. In other words, the instances which is likely to produce classifiers with high prediction accuracy (on labeled instances from the target

domain) and low local smoothness loss (on labeled and unlabeled instances) will be given higher weight, while other instances will be given lower weights. In this approach, more domain knowledge will be taken into account to re-weight source domain instances, making it less sensitive to the labeled instances and more reliable for the whole target domain. Roughly speaking, our new algorithm can be regard as an extention to both TrAdaBoost and semi-supervised boosting.

Instance-transfer learning and feature-representation-transfer learning are related to our work. The idea of instance-transfer learning [2,3,6] has been introduced above. For example, TrAdaBoost extends AdaBoost to the case of transfer learning. COILT algorithm [3] uses a learning strategy similar to Co-training. The essencial idea of feature-represenation-transfer learning is to find good feature representation across domains [7,8]. For example, TCA algorithm finds new feature representations in a Reproducing Kernel Hilbert Space [7]. [8] proposes to use sparse coding to learn higher-level features across domains.

The rest of the paper is organized as follows. The notations and formal problem statement are proposed in section 2. We propose the new algorithm in section 3. Extensive experiment results are presented in section 4. Finally, we conclude the paper with some discussion about future work in section 5.

2   Problem Statement and Notations

We focus on transfer learning problems with the same feature space for the two domains. We denote the feature space X as R^n, where n is dimension of the feature space. The class label set is shared across domains, which is $Y = \{-1, 1\}$.

We denote the source domain instance set as $S = \{(x_i, y_i)|x_i \in R^n, y_i \in Y, i = 1, 2, ..., m\}$, where x_i and y_i are the feature vector and the corresponding class label, and m is the number of source domain instances. The labeled target instance set is denoted as $T_l = \{(x_i, y_i)|i = m+1, m+2, ..., m+l\}$, and the unlabeled target instance set as $T_u = \{x_i|i = m+l+1, m+l+2, ..., m+l+u\}$, where l, u are the number of instances in T_l and T_u, respectively. Furthermore, l is assumed to be much less than m. In boosting algorithm, we denote the base classifier trained in t^{th} iteration as $f_t(x) : R^n \rightarrow [-1, 1]$. The objective is to train a classifier $F : R^n \rightarrow R$ to predict any instance x from the target domain.

3   Transfer Learning with Local Smoothness Regularizer

3.1   TrAdaBoost Algorithm and RegBoost Algorithm

TrAdaBoost Algorithm. In each iteration of TrAdaBoost, a base learner f_t is trained using the re-weighted instances. If a source domain instance is missclassified by f_t, it would be regarded as less relevant, and its weight will be decreased. On the contrast, if a target domain instance is wrongly predicted, its weight will be increased. The base classifiers are combined together similar to AdaBoost [1,2]. Note that the way TrAdaBoost re-weight the source domain instances is the Hedge(β) algorithm from [9]. As is stated in [2], the upper

bound of its generalization error is $\epsilon + O(\sqrt{\frac{Td_{vc}}{l}})$, where d_{vc} is the hypothesis space's VC-dimension. Since the value of l is small in transfer learning, the generalizeation error might be much larger than the empirical training error ϵ.

RegBoost Algorithm. AnyBoost is an abstract boosting algorithm [10]. From the perspective of functional analysis, boosting is an iterative process to optimization the total cost function $C_0(F) = \sum_{x_i \in Labeled} M(-y_i F(x_i))$, where M is a monotonically decreasing function [10]. Suppose the ensemble learner after k iterations is F_k, finding the next base learner f_{k+1} is eqauvallent to finding a new element f, such that $C(F_k + \epsilon f)$ decrease most rapidly.

RegBoost extends AnyBoost as follows [11]. First, defind pseudo-margin of each unlabeled instance as: $M_p(x) = M(|F(x)|)$ [12]. The total loss function is modified as: $C(F) = \sum_{x_i \in Labeled} \alpha_i M(y_i F(x_i)) + \sum_{x_i \in Unlabeled} \alpha_i M(|F(x_i)|)$, where α_i's are weight parameters. Second, defind pseudo-class for each unlabeled instance x_i: $y_i = sigh(F(x_i))$. In each iteration, the pseudo-class labels are treated equally as true labels, and the objective function to be maximize becomes:

$$- < \bigtriangledown C(F), f >= - \sum_{x_i \in Labeled \cup Unlabeled} \alpha_i y_i M^{'}(y_i F(x_i)). \qquad (1)$$

Furthermore, for each training instance x_i, an local smoothness regularizer $R_0(x_i)$ is introduced: $R_0(x_i) = \sum_{j:x_j \in Labeled \cup Unlabeled, j \neq i} exp(\frac{-||x_i - x_j||^2}{2\delta^2})c(-I_{ij})$, where I_{ij} indicating whether x_i and x_j share the same class (or pseudo-class) label, $W_{ij} = exp(\frac{-||x_i - x_j||^2}{2\delta^2})$ is an affinity measure, and $c(\gamma)$ is a monotonically decreasing function. The objective function for base learners becomes:

$$- < \bigtriangledown C(F), f > - \sum_{x_j \in Labeled \cup Unlabeled} \beta_j R_0(j). \qquad (2)$$

3.2   Transfer Learning with Local Smoothness Regularizer

Considering our motivation of introducing local smoothness, we want to improve TrAdaBoost in two folds: first, if an instance from the source domain is likely to increase the local smoothness loss of the base learner, its weight will be decreased; second, both the prediction accuracy the local smoothness loss would be used to evaluate the base learners. Specifically, the following strategy is adopted: for the source domain instances, the same Hedge(β) method is applied to reweight them. But for the target domain instances, some modification is adopted. Roughly speaking, we define new local smoothness regularizer for the target domain, and use RegBoost to take the place of AdaBoost. Furethermore, local smoothness is taken into consideration in evaluating base learners.

For the margin loss function, we use $M_0(t) = e^{-t}$. And the local smoothness regularizer is defined on the target domain instances:

$$R(x_i) = \sum_{m+1 \leqslant j \leqslant m+l+u, j \neq i} exp(\frac{-||x_i - x_j||^2}{2\delta^2})c(-I_{ij}), i = m+1, m+2, ..., m+l+u.$$

$$(3)$$

Here $c(\gamma)$ is defined as $c(\gamma) = M_0(\gamma) - 1$, the same as in [11].

Inserting M_0 and R_0 into (2), the objective function in each iteration is:

$$- \sum_{m+1 \leqslant i \leqslant m+l+u} \alpha_i y_i e^{-y_i F(x_i)} - \sum_{m+1 \leqslant i \leqslant m+l+u} \beta_i R(x_i). \qquad (4)$$

As in [11,12], divide (4) through by $- \sum_{i=m+1}^{m+l+u}(\alpha_i e^{-y_i F(x_i)} - \beta_i R(x_i))$, and the f which maximize (4) is equivalent to the f which minimize the following:

$$2 \sum_{y_i f(x_i)=-1} d_i - 2 \sum_{y_j f(x_j)=1} \frac{\beta_j R(x_j)}{\sum_{k=m+1}^{m+l+u}(\alpha_k e^{-y_k F(x_k)} - \beta_k R(x_k))} - 1, \qquad (5)$$

where $d_i = \frac{\alpha_i e^{-y_i F(x_i)} - \beta_i R(x_i))}{\sum_{k=m+1}^{m+l+u}(\alpha_k e^{-y_k F(x_k)} - \beta_k R(x_k))}$. From (5), f can be found as follows: each instance weight is updated as d_i, the base learning algorithm is applied to obtain f, and the error rate is modified and used to compute the weight of f.

Formally, the weight updating function is:

$$d_i = \frac{\alpha_i e^{-y_i F(x_i)} - \beta_i R(x_i))}{\sum_{i=m+1}^{m+l+u}(\alpha_i e^{-y_i F(x_i)} - \beta_i R(x_i))}, m+1 \leqslant i \leqslant m+l+u. \qquad (6)$$

And the error rate is modified as:

$$\epsilon' = \sum_{i:y_i f(x_i)=-1} d_j + \sum_{j:y_j f(x_j)=1} \frac{-\beta_k R(k)}{\sum_{k=m+1}^{m+l+u}(\alpha_k e^{-y_k F(x_k)} - \beta_k R(x_k))}. \qquad (7)$$

ϵ' is applied to compute the weight of the base learner:

$$w_f = 0.5 * ln(\frac{1 - \epsilon'}{\epsilon'}). \qquad (8)$$

We give the detailed algorithm(called Smooth-TrBoost for short) in Fig.1. There are two key steps in the algorithm. First, we construct three different classifiers to vote together to give pseudo-labels for each unlabeled target instances. Second, we will give lower initial weights to unlabeled target instances in the experiments. This is similar to that of ASSEMBLE [12].

4   Experiments

4.1   Data Sets and Comparision Methods

In the experiments, we take 2 popular text data sets to test the new algorithm, including 20 Newsgroups and Reuters-21758. All these data sets are organized in hierarchical structure, and we are able to construct data sets which have different underlying distributions but are closely related. Each time, we choose two of the top-level categories as the two class labels. There are some different sub-categories under both of these categories. In either of these top-level categories,

Smooth-TrBoost Algorithm

Input: source data: S; target data: T_l, T_u; Base Learner L;
Initialize 1: the initial weight vector D_1 for all the labeled and unlabeled instances
Initialize 2: pseudo-label for each instance in T_u(We use SVM($S \cup T_l$), SVM(T_l) and SVM(S) to vote for the pseudo-label)
Initialize 3: Initialize $F(x) := 0$, set all α_i's as 1, and all β_i's as the same value β

for $t = 1$ to T **do**
    1. Set $P_t(i) = \frac{D_t(i)}{\sum_{j=1}^{m+l+u} D_t(j)}$;
    2. Call Learner, and learn a classifier with S, T_l, T_u and the sample distribution P_t, and get back $h_t = L(S, T_l, T_u, P_t)$;
    3. if $t = 1$, set ϵ_t as the error rate of h_t, else set ϵ_t as in (7);
    4. Set the weight of h_t (i.e., w_t) as in (8), and update $F := F + w_t h_t$;
    5. Update pseudo-labels for instances in T_U: $y_i = F(x_i), i = m+l+1, ..., m+l+u$;
    6. Re-weight instances from target domain as in (6);
    7. Re-weight instances from source domain: if $h_t(x_i) \neq y_i$, set $D_t(i) = D_t(i)/(1 + \sqrt{2 \ln m/T})$;
end for
Output the final hypothesis for the target domain: $H(x) = \sum_{i=\lceil T/2 \rceil}^{T} w_i h_i(x)$

Fig. 1. Smooth-TrBoost Algorithm

Table 1. Data Summary

Data Set	No. of Features	Source Domain Size	Target Domain Size
comp.vs.rec	5681	2431	1951
comp.vs.sci	6773	2007	2373
comp.vs.talk	6418	2218	1837
rec.vs.sci	6935	1963	1992
rec.vs.talk	6203	1885	1761
sci.vs.talk	6390	1663	1939
orgs.vs.places	4415	1016	1046
orgs.vs.people	4771	1239	1210
people.vs.places	4562	1079	1080

data from some of the sub-categories are choosen as the target domain data, while data from other sub-categories are choosen as the source domain data. We use LingePipe[1] to process text files and transform them into word vectors. We summarize the information of these processed data sets in table.1.

We compare the new algorithm with several algorithms. First, two SVM classifiers are trained on different data: the classifier trained on T_l is denoted as SVMt, and the one trained on $T_l \cup S$ is denoted as SVMts. All the SVM classifiers are implemented using LibSVM[2]. Second, RegBoost is used for comparision. Finally,

---

[1] http://alias-i.com/lingpipe/
[2] http://www.csie.ntu.edu.tw/ cjlin/libsvm

we choose TrAdaBoost as baseline method. We use both SVM and TSVM [13] as base learners, and obtain two different TrAdaBoost classifiers, which are denoted as TrAda.SVM and TrAda.TSVM, respectively.

4.2 Experiment Results and Analysis

Performance Comparision. In the experiments, each data set from the target domain is split into 3 seperate sets to server as labeled training data set, unlabeled target data set and test data set, respectively. Denote the number of all the source domain instances as N_s, and the detail is as follows: first, about $0.01N_s$ instances are randomly selected to form the labeled training data set(i.e., as T_l, and $|T_l|$ is denoted as N_l), about $10N_s$ instances are also randomly selected to form the unlabeled training data set(i.e., T_u, and $|T_u|$ is denoted as N_u), and the rest of the instances in the target domain are put together to form the test data set(denoted as T_{test}). For each data set in table 1, the process above is repeated 20 times, and the average error rate of each algorithm is reported as its performance measure. For Smooth-TrBoost, we set the initial weights of each unlabeled instances to be $|N_l|/|N_u|$. Furthermore, we set $T = 45$, $\beta = 0.1$.

First, We compare Smooth-TrBoost with 3 different algorithms which do not use unlabeled instance for training, and plot the experiment results in Fig.2. The two transfer learning algorithms work better in almost all the data sets(except in people.vs.places data set). And comparing the two transfer learning methods, Smooth-TrBoost outperforms TrAda.SVM on all the 9 data sets. Roughly speaking, the improvement of Smooth-TrBoost over TrAda.SVM is at least 1.7%, and more than 5% in several cases.

Secondly, we depict the error rate curves of RegBoost, TrAda.TSVM and Smooth-TrBoost in Fig.3. All the algorithms use unlabeled instances in their training processes. The improvements of the two transfer learning algorithms over RegBoost are all more than 13% in 8 data sets. Comparing Smooth-TrBoost with TrAda.TSVM, Smooth-TrBoost improve the accuracy more than 2% on 5 data sets(i.e., comp.vs.rec, comp.vs.sci, comp.vs.talk, rec.vs.talk, sci.vs.talk, people.vs.places). The results confirm the efficiency of our algorithm.

Parameter Sensitivity Test. First, we select several data sets to test the sensitivity of β. In these experiments, we report the result of each single experiment, so that we can see exactly how the varying parameters affect the performance of Smooth-TrBoost. Specifically, each time, the data set is randomly splite into 3 sets, and for each splite, the algorithm is carried out 4 times, with β varying from 0.1 to 0.25(0.1, 0.15, 0.2, 0.25 respectively). The random splite is done 4 times for each data set. The results on orgs.vs.places data set are plot in Fig.4. It can be observed that, when β varies in $[0.1, 0.25]$, the different classifiers performance almost the same. To be exactly, the difference between their test error rates is not worse than 0.68%. Simliar results can be obtained from other data sets, but they are not depicted here due to space limitaions. These results demonstrate that, if the parameter of β is choosen properly, the performance of the Smooth-TrBoost algorithm would not be sensitive to slight change of pamareter vales.
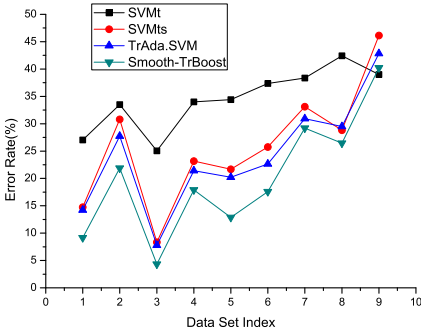
Fig. 2. Performance comparision (1)
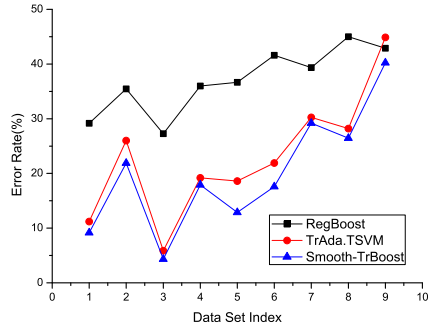
Fig. 3. Performance comparision (2)

Secondly, we vary the number of iterations T to study its sensitivity. 3 data sets are selected for experiments, including comp.vs.sci data set, orgs.vs.people data set and rec.vs.sci data set. Similar experiments are conducted with T varying from 15 to 45. For each data set, the experiments are repeated 20 times. We depict the average error rate in Fig.5. When T is not large enough, the increament of the number of iterations would improve the test accuracy of the classifiers. But after the value of T is large enough(e.g., $T \geq 30$ on rec.vs.sci data set), the increament of T would no longer improve the accuracy. The convergence of Smooth-TrBoost with respect to the value of T is similar to TrAdaBoost [2].
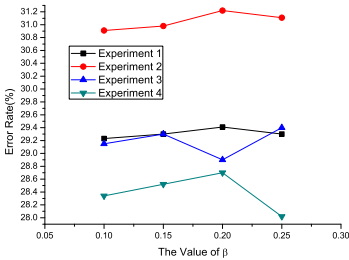
Fig. 4. Error rate with respect to β on orgs.vs.places data set
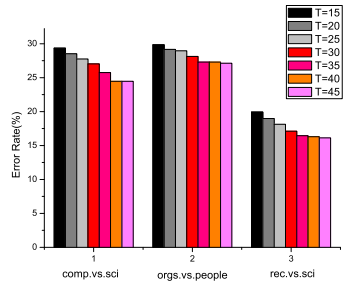
Fig. 5. Error rate with respect to the number of iterations

5   Conclusion

In this paper, we study the problem of instance-transfer learning. To bring more domain knowledge into the training process, we use unlabeled instances to help justifying the re-weighting of instances. Specifically, we propose to use local smoothness condition as additional constraints for the target classifier. Following the essential idea of semi-supervised boosting, local smoothness regularizer

is introduced to define new loss function, so that more domain knowledge is considered in the training process, making the classifiers more reliable for the target domain. Experiments on extensive text data sets also show promising results, and demonstrate that the new algorithms outperform competitive algorithms.

References

1. Pan, J.S., Yang, Q.: A Survey on Transfer Learning. IEEE Transations on Knowledge and Data Engineering 22(10), 1345–1359 (2010)
2. Dai, W., Yang, Q., Xue, G., Yu, Y.: Boosting for Transfer Learning. In: 24th International Conference on Machine Learning, pp. 193–200 (2007)
3. Shi, Y., Lan, Z., Liu, W., Bi, W.: Extending Semi-supervised Learning Mehthods for Inductive Transfer Learning. In: 9th IEEE International Conference on Data Mining, pp. 483–492 (2009)
4. Zhu, X.: Semi-supervised Learning Literature Survey. Technical report, Department of Computer Sciences, University of Wisconsin, Madison (2005)
5. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold Regularization: A Geometric Framework for Learning from Examples. Technical report, University of Michigan (2005)
6. Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset Shift in Machine Learning. MIT Press, Cambridge (2009)
7. Pan, J.S., Ni, X., Kwok, J.T., Yang, Q.: Domain Adaptation via Transfer Component Analysis. In: 21th International Joint Conference on Artificial Intelligence, pp. 1187–1192 (2009)
8. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: 24th International Conference on Machine Learning, pp. 759–766 (2007)
9. Freund, Y., Schapire, R.E.: A Decision-Theoretic Generalization of On-Line Learning and An Application to Boosting. Journal of Computer and System Science 55, 119–139 (1997)
10. Mason, L., Bartlett, P., Baxter, J., Frean, M.: Functional Gradient Techniques for Combining Hypotheses. In: Advances in Large Margin Classifiers. MIT Press, Cambridge (2000)
11. Chen, K., Wang, S.: Regularized Boost for Semi-Supervised Learning. In: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge (2007)
12. Bennett, K., Demiriz, A., Maclin, R.: Exploiting Unlabeled Data in Ensemble Methods. In: 8th International Conference on Knowledge Discovery and Data Mining, pp. 289–296 (2002)
13. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In: 16th International Conference on Machine Learning, pp. 200–209 (1999)

Scalable Complex Event Processing
on Top of MapReduce

Jiaxue Yang, Yu Gu, Yubin Bao, and Ge Yu

Northeastern University, China
Yang.Jiaxue.Micheal@gmail.com, {guyu,baoyubin,yuge}@ise.neu.edu.cn

Abstract. In this paper, we propose a complex event processing framework on top of MapReduce, which may be widely used in many fields, such as the RFID monitoring and tracking, the intrusion detection and so on. In our framework, data collectors collect events and upload them to distributed file systems asynchronously. Then the MapReduce programming model is utilized to detect and identify events in parallel. Meanwhile, our framework also supports continuous queries over event streams by the cache mechanism. In order to reduce the delay of detecting and processing events, we replace the merge-sort phase in MapReduce tasks with hybrid sort. Also, the results can be responded in the real-time manner to users using the feedback mechanism. The feasibility and efficiency of our proposed framework are verified by the experiments.

1 Introduction

With the rapid development of various monitoring devices, simple event processing can not meet the requirements of continuous tracking and decision-making. Therefore, real-time analysis and processing over a series of events are desired in many applications. In recent years, complex event processing over streams is becoming a significant technique which needs to be urgently developed.

Most of the existing complex event processing *(CEP)* systems are centralized which transfer events collected by devices to a single node. The node utilizes the sliding window and automata model to identify and match events. Obviously, their processing ability and memory capacity are very limited. When facing massive source data, they may break down frequently. Some distributed systems have solved the problem to some extent, but fail to maintain the scalability. MapReduce[1,2] based systems are famous for the excellent scalability and flexibility, which offer the potential chance for the data-intensive event processing.

However, it is not feasible and efficient to apply the available MapReduce framework directly to the CEP scenarios. MapReduce framework was designed for online analytical processing. Therefore, there are mainly four challenges to adapt to the framework. First of all, because original events are often collected continuously, we need to solve the problem of storing data stream. Second, due to the high I/O cost, the merge-sort phase in MapReduce tasks is not suitable for many real-time CEP scenarios. And then, we always want to get results early

and gradually instead of simultaneously in massive source data. At last, how to cache partial matching results in the stream processing is also a problem.

In order to solve the potential challenges, this paper proposes a novel framework on top of MapReduce to support efficient complex event processing while retaining high scalability and fault-tolerance. By modifying the Hadoop[3,4], we implement a framework, which mainly consists of the event stream storage module and the event stream processing module. Specially, we design an uploading architecture to push the data stream to distributed file system *(DFS)*. Furthermore, our framework design a hybrid sort function instead of merge-sort phase. And then, we have improved the basic MapReduce framework by introducing an input cache for reducer to support continuous queries over streams, and by utilizing the feedback mechanism for users to get some results more quickly.

The remainder of this paper is organized as follows: The related work is described in Section 2. In Section 3, we introduce the event stream storage module. Section 4 describes how to execute a query in our framework and specifies the improvements on MapReduce. The experimental studies are presented in Section 5. Finally, the work is concluded in Section 6.

2   Related Work

The centralized CEP systems have been intensively studied in recently years. SASE[5] first proposes the CEP problem and optimizes the automata model for the efficient correlation. Furthermore, some systems[6,7] are designed for specified application scenarios of CEP. Also, some researchers have proposed some distributed CEP systems. For example, [8] introduces a distributed CEP system and the query rewriting and distribution schemes are proposed.

MapReduce framework is proposed to support parallel analysis and computing. In recent years, modifying MapReduce framework for different scenarios has become a hot topic. Some researchers have proposed a new type of system named Hadoop++[9], which proposes new index and join techniques, namely Trojan Index and Trojan Join to improve runtimes of MapReduce jobs. MRShare[10] transforms a batch of queries into a new one that will be executed more efficiently, by merging jobs into groups and evaluating each group as a single query. It provides a solution that derives the optimal grouping of queries. HaLoop[11] not only extends MapReduce with programming support, but also improves efficiency by the task scheduler loop-aware and various caching mechanisms. However, all these frameworks can not support CEP applications.

3   Event Stream Storage Module

3.1   Event Model

Event object could be expressed as a triple factor group (ID, Timestamp, Event-Type). Among them, the *ID* of an event is a unique identification. We can use

it to distinguish persons or goods from others. *ID* may be the article number in the supermarket application. *Timestamp* is the time when a simple event is collected(e.g. detected by a RFID reader). *EventType* means the event's category(e.g. it can be classified according to the *ID* or location of the RFID reader).

3.2   Centralized Upload

At first, we design a master-slave mode in the event stream storage module, including a master node and some data collectors as slave nodes shown in figure 1. Data collectors are responsible for collecting event information in real-time and sending them to the master node immediately. There is no need to have very strong processing ability and memory capacity in data collectors.
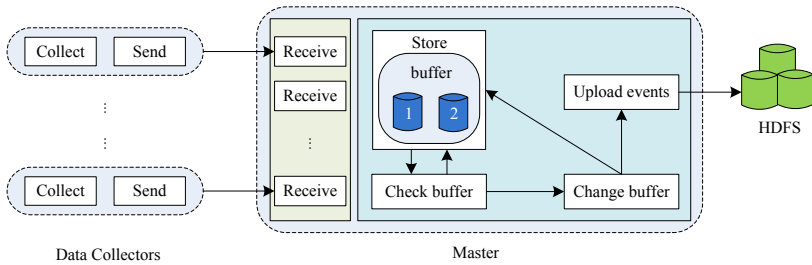
Fig. 1. Centralized upload architecture

We have adopted double buffer technology on the master node, which uses buffer *one* and buffer *two* alternately to receive events and preserve them. It is assumed that our framework is receiving events utilizing buffer *one* at T_0. At the moment of $T_1(T_1 > T_0)$, when buffer *one* is full of events or the number of them has exceeded the user-defined threshold, our framework will start to utilize buffer *two* instead of buffer *one*. Meanwhile, it will create a thread to upload all the events in buffer *one* to DFS. At $T_2(T_2 > T_1)$, events in buffer *two* have met the same uploading situation, and the processing method is similar to T_1. Taking turns using two buffers could guarantee that there is no need to add a synchronous lock to the buffer while uploading events in a batch.

3.3   Distributed Upload

If we only use a node as the master node to receive and upload events, it will become a bottleneck of our framework. The processing ability and memory capacity of a single node is very limited. To avoid that problem, we design and implement three-layer structure: data collection layer, event storage layer and path coordination layer. Our structure allows uploading events asynchronously in a batch. Path coordination layer usually stores the directory for uploading. Every node in the event storage layer is the same with the master node mentioned in the section 3.2. When any buffer is full, it will get a directory from the

path coordination layer and upload events to DFS immediately. We can increase some nodes to improve the scalability of our framework. Because of that, our framework could execute a query processing based on massive source data.

4   Event Stream Processing Module

Our framework will return the results with very low latency. The architecture of our event stream processing module is shown as figure 2. It is divided into four layers: application layer, parsing layer, computation layer and storage layer. In the application layer, we support different applications, such as SEQ() and COUNT(). As soon as receiving some queries, our framework will classify and parse them to the corresponding deterministic finite automatons*(DFAs)* by utilizing the query parsing engine in the parsing layer. And then, it will transfer a DFA to the MapReduce task and execute it immediately. Finally, MapReduce task will read events from the storage layer and results will be sent to users in real-time. We use DFS, RDBMS and local file systems as our storage layer.
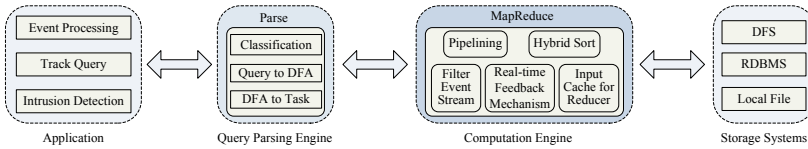
Fig. 2. Event stream processing module architecture

4.1   MapReduce Task

Different queries may have similar executions with each other in our framework. Map task is responsible for filtering some events which have no relationship with the query. Reduce task usually utilizes DFA to identify whether the result has been found or not. As an example, SEQ(A, B, C) is shown as figure 3.

Every file in DFS has many events. First, input files should be changed into event objects through the InputFormat phase. Map task is to filter events to reduce the cost of the sort and group phase. The output of map task is like *(ID + Timestamp, Event)*. After map tasks, the partition phase will begin. It uses a hash function only on the attribute *ID* to ensure that the events with the same *ID* will be processed in one reduce task. Before the reduce phase, our framework should sort the input data according to the key which has two attributes *ID* and *Timestamp*. And the group function will be only used on the attribute *ID* after sorting. Reduce task is to match and identify events gradually for each group. When the results are matched, our framework will return them to users immediately. There is no output in each reduce task, so our framework skips the OutputFormat phase. We also design the hybrid sort, feedback mechanism and input cache for the reducer. We will describe them later.
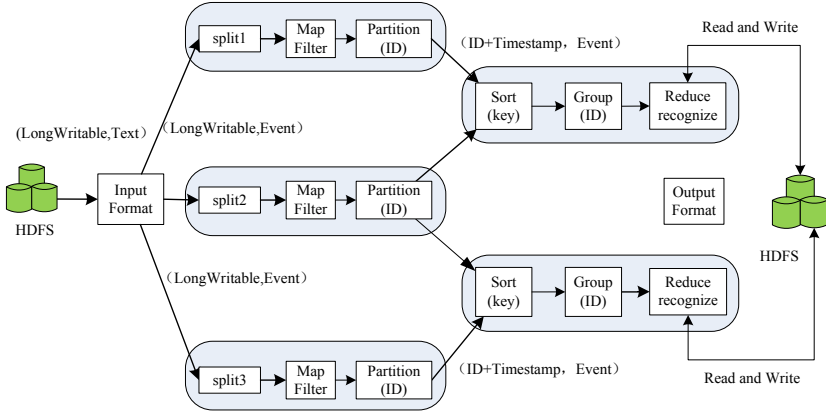
Fig. 3. MapReduce task of SEQ(A, B, C)

4.2 Hybrid Sort

We have to sort all the events by attribute *Timestamp* and group them by attribute *ID* in some query processings. After that, our system could begin to match and identify events. The basic MapReduce uses merge-sort in the sort phase. Although it can meet our requirements, merge-sort may cause very high I/O costs. It wastes too much time and the latency has to be delayed. To solve this problem, we have changed the MapReduce framework by replacing merge-sort with the hybrid sort. Hybrid sort is shown as figure 4, which is executed on the reduce node. And it blends a hash function and a merge-sort phase.

The output of map task is in the format of *(key, value)*. Key contains *ID* and *Timestamp* information. We will utilize the hash function on *ID* so that events with different *ID* could be set in different buckets. If the number of events in a bucket is beyond the memory capacity, we will save all the information of the bucket to local disks. For every bucket, our system uses merge-sort to sort events. After that, the events in a bucket have already been in the order of *Timestamp* and they will be processed as the same group by a reduce task.

When the data exceeds the memory capacity, the I/O cost of merge-sort may be as (1). F is the merge factor and B means the memory size. They could be configured in the files. N means the size of data set as the reduce input.

$$G(n) = (\frac{B^2}{2F(F-1)}n^2 + \frac{3B}{2}n - \frac{F^2}{2(F-1)}) \cdot B \qquad (1)$$

$$F(n) = \frac{2n}{B} + \sum_{i=0} \alpha_i \cdot G(n_i) \qquad (\sum_{i=0} n_i = n) \qquad (2)$$

The I/O cost model of our hybrid sort method is shown as (2). $\frac{2n}{B}$ is the cost of grouping by the hash function. The total I/O cost of merge is the sum of each bucket's I/O cost, which is $\sum \alpha_i \cdot G(n_i)$. Among them, n_i means the size

of events in bucket i and α_i is a binary function. If $n_i > B$, α_i equals to 1. Otherwise α_i equals to 0. Because events of each bucket are often less than the memory size, α_i usually equals to 0, and thus hybrid sort has less I/O cost.
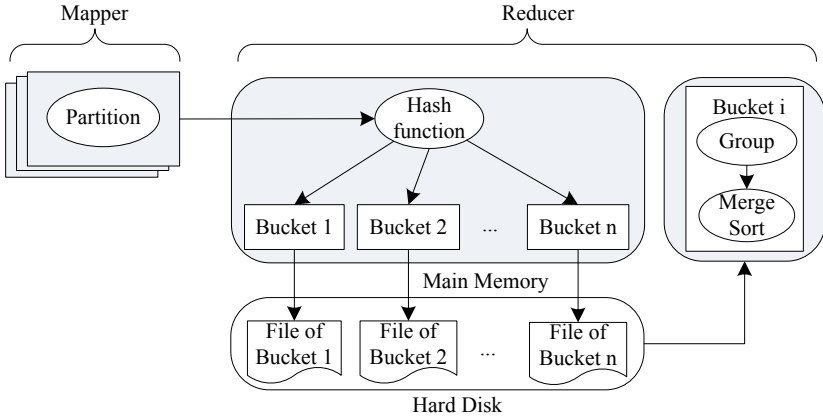
Fig. 4. Hybird sort

4.3   Feedback Mechanism

The basic MapReduce framework will save the output to DFS, so that our results have also to be stored into DFS. Users need to write another function to look up in all the results together. Our system utilizes the message mechanism and multi-thread technology in the reduce function to send results to users directly. Once matched, the results will be sent immediately. It makes users get the results gradually and users get them earlier by our feedback mechanism.

4.4   Input Cache for Reducer

We design and implement an input cache for reducer to support continuous query over event streams. When the next events have arrived, we only need to maintain the results incrementally, which are saved by the last batch processing. There is no need to scan all the events from the beginning to match the results. In order to keep the good fault-tolerance, we utilize the distributed file system or local disks as our cache mechanism. At the end of every MapReduce task, our framework will save the set of partial matched status of the DFAs to the input cache for reducer. When the next reduce task begins, it will read the input cache for reducer firstly, and then maintain the status incrementally.

5   Evaluation Performance

All the experiments are run on the cluster, which is composed of eleven nodes, including one master and ten slaves. Every node contains 2.00GHz Intel Xeon

CPU, 2GB RAM, 75GB SATA disks at 7,200rpm speed. All nodes run Hadoop-0.19.2 with the default parameters on Red Hat Enterprise Linux Server release 5.4 and are connected by gigabit Ethernet.

(a) MapReduce tasks

(b) Real-time feedback

(c) Filter technique

(d) Continuous query
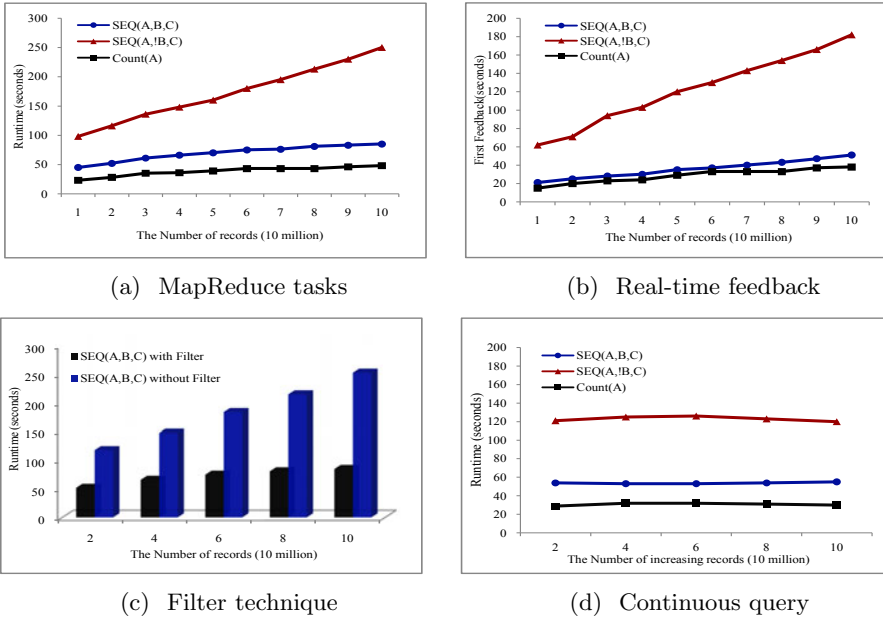
Fig. 5. Evaluation performance with the simulated data set

Our experiments are performed on 2.7GB simulated data set including 10^8 records *(ID,Timestamp,EventType)*. We have simulated some applications about the theft in the supermarket, the monitor in the road network and the aggregation of clicks in the website, where SEQ(A,!B,C),SEQ(A,B,C) and Count(A) are separately used. We also have designed extensive experiments about runtime of MapReduce tasks on different records, filter technique, continuous query, and first feedback time. The experimental results are shown as figure 5.

According to the experiments, our system maintains the good scalability of Hadoop. As figure 5(a) shows, when the events are increasing in ten million level, the runtime of each query will not grow very rapidly. Even if we inquire on hundreds of millions events, Count() could return the results in one minute and SEQ() in five minutes. In figure 5(b), we improve our framework by the feedback mechanism. It will return all the results to users directly and asynchronously. Comparing with figure 5(a), users can get the first result of each level more than 50% earlier. Our framework firstly filter events which have nothing to do with queries. The selectivity of our data set is about 3.8% of Count(A) and 11.5% of SEQ(A,B,C) in figure 5(c). It shows that the runtime of the MapReduce tasks will reduce to 40% using filters, because we have fewer records to sort and match. It will have fewer I/O costs if memory usage is not beyond the buffer size. In

figure 5(d), our framework only matches and detects the newly coming twenty million events incrementally by utilizing the input cache for reducer. It is obvious that our performance has been improved a lot comparing to figure 5(a).

6   Conclusion

This paper has put forward a novel framework on top of MapReduce to support efficient complex event processing while remaining high scalability. It solves the problem of efficiently processing massive source data of the various complex event processing applications. We have designed an uploading architecture to support massive source data storage. The basic MapReduce framework has been improved by the hybrid sort, feedback mechanism and input cache for reducer. The experimental results verify our framework has good performance and scalability in CEP query processing. In the future, we will further design the effective share mechanism to optimize multi-users shared queries.

References

 1. Jeffrey, D., Sanjay, G.: Mapreduce: Simplied data processing on large clusters. In: OSDI (2004)
 2. Jeffrey, D., Sanjay, G.: Mapreduce: a fiexible data processing tool. Communications of the ACM (2010)
 3. Tom, W.: Hadoop: The Definitive Guide. O'Reilly, Yahoo! Press (2009)
 4. hadoop (2011), `http://hadoop.apache.org/`
 5. Eugene, W., Yanlei, D., Shariq, R.: High-Performance Complex Event Processing over Streams. In: SIGMOD (2006)
 6. Kyumars, S.E., Tahmineh, S., Peter, M.F.: Changing Flights in Mid-air: A Model for Safely Modifying Continuous Queries. In: SIGMOD (2011)
 7. Chun, C., Feng, L., Beng, C.O.: TI: An Efficient Indexing Mechanism for Real-Time Search on Tweets. In: SIGMOD (2011)
 8. Nicholas, P., Matteo, M., Peter, P.: Distributed Complex Event Processing with Query Rewriting. In: DEBS 2009 (2009)
 9. Jens, D., Jorge-Arnulfo, Q., Alekh, J.: Hadoop++: Making a Yellow Elephant Run Like a Cheetah. In: VLDB (2010)
10. Tomasz, N., Michalis, P., Chaitanya, M.: MRShare: Sharing Across Multiple Queries in MapReduce. In: VLDB (2010)
11. Yingyi, B., Bill, H., Magdalena, B.: HaLoop: Efficient Iterative Data Processing on Large Clusters. In: VLDB (2010)

Learning to Recommend Based on Slope One Strategy

Yongqiang Wang, Liang Yin, Bing Cheng, and Yong Yu

Computer Science Department,
Shanghai Jiao Tong University,
Dongchuan Road, Minhang District, Shanghai, China
{wangyq,yinla,chengb,yyu}@apex.sjtu.edu.cn

Abstract. Recommendation systems provide us a promising approach to deal with the information overload problem. Collaborative filtering is the key technology in these systems. In the past decades, model-based and memory-based methods have been the main research areas of collaborative filtering. Empirically, model-based methods may achieve higher prediction accuracy than memory-based methods. On the other side, memory-based methods (e.g. slope one algorithm) provide a concise and intuitive justification for the computed predictions. In order to take advantages of both model-based and memory-based methods, we propose a new approach by introducing the idea of machine learning to slope one algorithm. Several strategies are presented in this paper to catch this goal. Experiments on the MovieLens dataset show that our approach achieves great improvement of prediction accuracy.

Keywords: Collaborative filtering, recommendation, machine learning, slope one.

1 Introduction

Confronted with the rapid growth in the amount of information, how to find and choose the information of interest is a great challenge. Recommendation systems provide us a promising approach to deal with the information overload problem. Due to their great performance, currently recommendation systems have been used in a wide range of areas and applications, such as recommendations of CDs, movies, books and web pages.

A recommendation system estimates the responses of a user for new items, based on historical information of the user stored in the system, and recommends novel and original items to this user [1,2]. Many technologies have been developed for recommendation systems. Collaborative filtering (CF) approaches are the most popular and efficient. CF approaches can be grouped into two categories: memory-based and model-based. Empirically, model-based methods may achieve higher prediction accuracy than memory-based methods. On the other side, memory-based methods (e.g. slope one algorithm) provide a concise and intuitive justification for the computed predictions.

In this paper, we try to capture the advantages (accuracy and justifiability) of both model-based and memory-based CF approaches. Due to the no-paraphrase of model-based method, we do not directly exploit the model-based approaches. Instead, we consider the common idea of the model-based approaches: minimizing the whole prediction error by learning the parameters. On the other side, we take slope one algorithm which is an outstanding memory-based CF approach into consideration. To our best knowledge, this is the first attempt towards integrating slope one predictor with the idea of machine learning.

The rest of the paper is organized as follows. In the next section, we discuss related work. Then we describe approaches in section 3. Section 4 shows the results of our experiments and presents the experiment comparison of different approaches with discussion. At last, section 5 concludes the paper.

2   Related Work

In this section we briefly present the related work and literatures of recommendation systems and collaborative filtering.

Content-based [7] methods recommend items in line with user's interests by understanding and analyzing both the interests of users and the content of items. Such systems based purely on content generally suffer from the problems of limited content analysis and over-specialization.

User-based [2] approaches are first proposed in collaborative filtering. The inherent assumption is that users like the items which are liked by their similar users (neighbors). In some applications, compared to the rapid growth of the number of users, the number of items may increase slowly, so in order to reduce the similarity computation, item-based [2,4] CF approaches are proposed. Slope one algorithm which is considered as the simplest form of item-based approaches was introduced in [6]. Slope one algorithm proposes a predictor of the simple form $f(x) = x + b$ to indicate the relation of any pair of items.

The CF techniques, especially the model-based approaches enjoyed a rapid development since October 2006 when the Netflix Prize competition [8] started. Singular value decomposition(SVD) methods, such as RSVD, NSVD [9], SVD++ [3,10], were proposed during that period. SVD methods are the most efficient of the model-based approaches.

In this paper we propose some learning-based slope one algorithms in order to gain both the high accuracy and the justifiability.

3   CF Approaches

This section detailedly presents the baseline slope one algorithms and our proposed learning-based slope one algorithms.

3.1   Notation

We briefly introduce the common notations used in the following presentation. In this paper we use u or v to indicate the user and i or j to indicate the item. The

rating of a given user u for the item i is expressed as r_{ui}, and the corresponding prediction is denoted as p_{ui}. The (u, i) pairs for which the ratings are known are stored in the set K. The set of all the items rated by user u is $S(u)$. $|S(u)|$ is the number of the items in the set. We use dev_{ij} to indicate the deviation of item i respect to item j. If no special instructions, dev_{ij} is the negative of dev_{ji}.

3.2   The Slope One Approach

The slope one algorithm is based on the intuitive principle of a "popular differential"(deviation) between items for users [6]. For a simple example, consider Table 1, item I gets 2 from user A, while item J gets 3, and user B gives a rating of 4 to item I, then what's the rating of user B for item J? By the slope one approach, we can calculate the deviation of item J respect to item I from user A: $dev_{JI} = 3 - 2 = 1$. Then we can infer the rating of user B for item J: $4 + dev_{JI} = 5$.

Table 1. A simple example to illustrate the slope one approach

	Item I	Item J
User A	2	3
User B	4	? = 4+(3-2)

Formally, the slope one algorithm exploits the predictor of the form $f(x) = x + b$ to indicate the relation of the item pair, where b is a constant deviation and x is a variable representing rating value. So the formal definition of dev_{ij} can be given here. Given two items i and j ($i \neq j$), the deviation of item i respect to j is as follow:

$$dev_{ij} = \frac{\sum_{u \in S_{ij}} (r_{ui} - r_{uj})}{|S_{ij}|}. \tag{1}$$

Here S_{ij} denotes the set of users who both rate item i and item j, and $|S_{ij}|$ is the number of the users in the set. All deviations construct a deviation matrix which is a skew-symmetric matrix.

With the deviation matrix constructed, it is easy to make predictions by using the slope one algorithm. Regarding $r_{uj} + dev_{ij}$ as a prediction for rating r_{ui} given r_{uj}, a more reasonable predictor might be the average of all such predictions:

$$p_{ui} = \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + dev_{ij})}{|S(u) - \{i\}|}. \tag{2}$$

Here $S(u) - \{i\}$ represents the set of all items rated by user u except item i.

3.3   The Weighted Slope One Approach

In the above approach, we average the predictions calculated from each individual items to generate a final prediction. Actually they have different contributions to the final prediction. Simply using the average is not a good solution. Hence, a better predictor (i.e. the weighted slope one approach) was proposed as follow:

$$p_{ui} = \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + dev_{ij}) * c_{ij}}{\sum_{j \in S(u) - \{i\}} c_{ij}} . \tag{3}$$

Where c_{ij} in the equation is the number of users who both rate item i and item j. The bigger the c_{ij}, the more reliable the corresponding prediction (i.e. $r_{uj} + dev_{ij}$).

3.4   The Learned Slope One Approach

In our Learned Slope One Approach, we introduce the idea of machine learning from model-based algorithms into the traditional slope one predictor. We straightforwardly extend the traditional slope one predictor of the simple form: $f(x) = x + b$.

Because of the good performance of the weighted slope one approach, a natural idea to achieve higher accuracy is to find better weights. Combining with the idea of machine learning, we propose the learned slope one approach. The equation of prediction is similar to the previous one:

$$p_{ui} = \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + dev_{ij}) * w_{ij}}{|S(u) - \{i\}|} . \tag{4}$$

Here w_{ij} is adjusted through the training period. In our setting, w_{ij} is initialized to 1. Under this initialization, the base slope one predictor is the initial state of our approach. In order to learn the weight w_{ij}, we can solve the least squares problem:

$$\min_{w_*} \sum_{(u,i) \in K} (r_{ui} - \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + dev_{ij}) * w_{ij}}{|S(u) - \{i\}|})^2 . \tag{5}$$

A simple gradient descent method can be applied to solve the above problem. We first define the prediction error $e_{ui} = r_{ui} - p_{ui}$. Then for each rating r_{ui} in the training set, we update the parameters with the gradient descent technology:

- $\forall j \in S(u) - \{i\}$:

$$w_{ij} \leftarrow w_{ij} + \gamma * (e_{ui} * \frac{r_{uj} + dev_{ij}}{|S(u) - \{i\}|}) .$$

The parameter γ (learning rate) is determined by cross-validation.

3.5   The Biased Learned Slope One Approach

In general, user and item biases exist in the user-item rating matrix. Obviously, the learned slope one approach above does not consider the effects of these biases.

Currently, these biases are usually an indispensable part of model-based algorithms. The model which purely consists of these biases is dubbed as baseline estimator. The baseline estimate of the unknown rating r_{ui} is denoted as b_{ui} as follow:

$$b_{ui} = \mu + b_u + b_i \,. \tag{6}$$

Where the parameter μ is a constant value, the average of all ratings, and b_u denotes the user effect offset, while b_i denotes item effect offset.

Based on this baseline estimator, we can develop our learned slope one approach further:

$$p_{ui} = \mu + b_u + b_i + \frac{\sum_{j \in S(u)-\{i\}} (r_{uj} + dev_{ij} - b_{ui}) * w_{ij}}{\sqrt{|S(u) - \{i\}|}} \,. \tag{7}$$

Here, b_u, b_i and w_{ij} are parameters which are learned by gradient descent, while b_{ui} remains a constant value which is calculate by (6). In this formula, we avoid overemphasizing the difference between the heavy raters and the light raters via the normalization. We can learn the parameters by solving the following formula:

$$\min_{b_*, w_*} \sum_{(u,i) \in K} (r_{ui} - \mu - b_u - b_i - \frac{\sum_{j \in S(u)-\{i\}} (r_{uj} + dev_{ij} - b_{ui}) * w_{ij}}{\sqrt{|S(u) - \{i\}|}})^2$$
$$+ \beta * (\sum_u b_u^2 + \sum_i b_i^2 + \sum_i \sum_j w_{ij}^2) \,. \tag{8}$$

For each rating r_{ui}, we update the parameters as follow:

- $b_u \leftarrow b_u + \gamma * (e_{ui} - \beta * b_u)\,,$
- $b_i \leftarrow b_i + \gamma * (e_{ui} - \beta * b_i)\,,$
- $\forall j \in S(u) - \{i\} :$

$$w_{ij} \leftarrow w_{ij} + \gamma * (e_{ui} * \frac{r_{uj} + dev_{ij} - b_{ui}}{\sqrt{|S(u) - \{i\}|}} - \beta * w_{ij})\,.$$

3.6 The Biased Learned Slope One Approach with Deviation Learned

In the two previous learning-based approaches we mainly adjust the weights to optimize the model. In this section we further apply the leaning process on the deviation parameters. Here we still adopt (7) as the prediction function, where the parameter dev_{ij} is adjusted through training process.

In order to learn the parameters, we can solve the least square problem as follow:

$$\min_{b_*, w_*, dev_*} \sum_{(u,i) \in K} (r_{ui} - \mu - b_u - b_i - \frac{\sum_{j \in S(u)-\{i\}} (r_{uj} + dev_{ij} - b_{ui}) * w_{ij}}{\sqrt{|S(u) - \{i\}|}})^2$$
$$+ \beta * (\sum_u b_u^2 + \sum_i b_i^2 + \sum_i \sum_j w_{ij}^2) \,.$$

$$\tag{9}$$

The difference between (8) and (9) is that dev_{ij} here is also an optimization parameter. Through our experiments, we find that better performance can be achieved when dev_{ij} does not appear in the regularization part.

We update the parameters as follow:

- $b_u \leftarrow b_u + \gamma * (e_{ui} - \beta * b_u)$,
- $b_i \leftarrow b_i + \gamma * (e_{ui} - \beta * b_i)$,
- $\forall j \in S(u) - \{i\}$:

$$w_{ij} \leftarrow w_{ij} + \gamma * (e_{ui} * \frac{r_{uj} + dev_{ij} - b_{ui}}{\sqrt{|S(u) - \{i\}|}} - \beta * w_{ij}),$$

$$dev_{ij} \leftarrow dev_{ij} + \gamma * \frac{e_{ui} * w_{ij}}{\sqrt{|S(u) - \{i\}|}}.$$

4   Experiments and Evaluation

4.1   Dataset

Our experiments are based on the MovieLens dataset which is provided by the GroupLens Research Project at the University of Minnesota and can be downloaded freely. This dataset is widely used in research areas about collaborative filtering, such as [4,6]. The dataset contains 100,000 ratings from 943 users on 1682 movies, where the rating value ranges from 1 to 5.

4.2   Setup and Evaluation Metrics

The full dataset of the ratings is divided into two parts: training set and testing set. In our experiments, 80% of the dataset is used as training data and the rest 20% as testing data. Five-fold cross-validation is exploited to determine the parameters γ and β. In the experiments, we set $\gamma = 0.01$ and $\beta = 0.03$.

In this paper we calculate Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to measure the prediction accuracy. The formal definition of MAE is as follow:

$$MAE = \frac{\sum_{(u,i)\in T} |r_{ui} - p_{ui}|}{|T|}, \tag{10}$$

where T denotes the testing set, and $|T|$ is the number of elements in T. The formula of RMSE is as follow:

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in T} (r_{ui} - p_{ui})^2}{|T|}}. \tag{11}$$

4.3   Method Evaluation and Comparison

We evaluate all the approaches introduced in section 3, namely the Slope One Approach (SO), the Weighted Slope One Approach (WSO), the Learned Slope One Approach (LSO), the Biased Learned Slope One Approach (BLSO) and the Biased Learned Slope One Approach with Deviation Learned (BLSOD).

SO and WSO approaches are traditional methods with no learning process. We initialize w_{ij} to 1 for LSO. And for BLSO and BLSOD, b_u, b_i and w_{ij} are all initialized to 0. The experiments show that complicated initialization does not bring large improvement in BLSO approach.

The results of MAE and RMSE of these approaches are shown in Fig.1 and Fig.2 respectively. Through our experiments, it is obvious that all the learning-based slope one algorithms (i.e. LSO, BLSO and BLSOD) outperform the baseline. The best results achieved by these approaches are illustrated in Table 2.
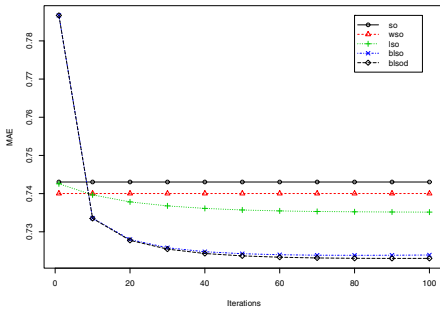
Fig. 1. Comparison of MAE
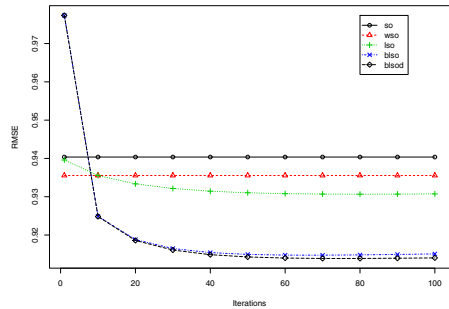
Fig. 2. Comparison of RMSE

From Table 2 we can observe that the BLSOD approach performs only slightly better than BLSO. We consider that there may be two reasons: 1. dev_{ij} is updated according to the w_{ij} which is usually very small, so the change of dev_{ij} is too slight to affect the result; 2. the influence of parameter dev_{ij} is weaker than w_{ij} in the prediction.

Table 2. The best result achieved by each approach. Result in bold is the best under the corresponding evaluation metric.

Method	MAE	RMSE
so	0.7430	0.9403
wso	0.7400	0.9355
lso	0.7351	0.9306
blso	0.7239	0.9147
blsod	**0.7230**	**0.9137**

5   Conclusion and Future Work

In our work, we aim to achieve both the justifiability and high accuracy in recommendation systems. In order to reach this goal, we introduce the idea of machine learning into the slope one approach which is a very simple and efficient memory-based approache. The experiments on the MovieLens dataset show that all these learning-based slope one methods exhibit their great performance. The bias information of users' ratings is demonstrated to be very useful as well. Compared with the Weighted Slope One approach (WSO), our best result achieves an considerable improvement (2.3% for MAE, 2.8% for RMSE respectively).

We will focus on two aspects in our future work. First, these learning-based slope one algorithms can be integrated with other model-based algorithms to achieve higher accuracy. Second, we will try to develop methods to make the recommendations more understandable and justifiable.

References

1. Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.): Recommender Systems Handbook. Springer (2011)
2. Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In: Recommender Systems Handbook, pp. 107–144 (2011)
3. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD 2008: Proceeding of the 14th ACM SIGKDD, pp. 426–434. ACM, New York (2008)
4. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW 2001: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295. ACM, New York (2001)
5. Koren, Y., Bell, R.M.: Advances in collaborative filtering. In: Recommender Systems Handbook, pp. 145–186 (2011)
6. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: Proceedings of the SIAM Data Mining Conference (2005)
7. Lops, P., Gemmis, M., Semeraro, G.: Content-based Recommender Systems: State of the Art and Trends. In: Recommender Systems Handbook, pp. 73–105 (2011)
8. Bennet, J., Lanning, S.: The Netflix Prize. In: KDD Cup and Workshop (2007)
9. Paterek, A.: Improving Regularized Singular Value Decomposition for Collaborative Filtering. In: The Proc. KDD Cup and Workshop (2007)
10. Koren, Y.: Factor in the Neighbors: Scalable and Accurate Collaborative Filtering. ACM Trans. Knowl. Discov. Data 4(1), 1–24 (2010)

Dataflow Optimization for Service-Oriented Applications

Peng Zhang[2,3], Guiling Wang[1], and Yanbo Han[1]

[1] North China University of Technology, Beijing, 100041, China
[2] Graduate University of Chinese Academy of Sciences, 100049, Beijing, China
[3] Institute of Computing Technology, Chinese Academy of Sciences, 100190, Beijing, China
{zhangpeng,wangguiling}@software.ict.ac.cn, yhan@ict.ac.cn

Abstract. Service-oriented architecture suits well for supporting large-scale cross-organizational business collaborations that can be commonly found in emergency management, supply chain management and healthcare management. However, in the applications, users usually have to make decisions within tight time frames and are overwhelmed with a large amount of SOAP data transfers. In this paper, a "by reference" data transfer strategy is proposed. The "by reference" data transfer strategy uses integration data structures to enable the elimination of the SOAP serialization and de-serialization within the service invocation by the exchange of their references. By integrating the "by reference" data transfer strategy, we can expect efficiency improvements in data transfer. Experiments and analysis supported our efforts.

Keywords: data flow, integration data structure, SOAP, data transfer.

1 Introduction

Service-Oriented Architecture (SOA) has become a commonly used paradigm for designing and building software-intensive applications which are composed of a number of loosely coupled distributed services. SOA promises to help create flexible dynamic business processes and agile applications that may span organizations and computing platforms. For emergency management, supply chain management and healthcare management applications, users usually have to make decisions within tight time frames and are overwhelmed with a large amount of data transfers, which are caused by the data transfer protocol called SOAP (Simple Object Access Protocol): Embedding massive structured data sets in a SOAP message is not a good solution from the performance perspective because of extensive de-serialization at the end-points. This paper has the following specific features:

1)   "By reference" data transfer. Each service description associated with the integration data structures, which carry the data aspects of functional Web services, enables the elimination of the SOAP serialization and de-serialization within the Web service invocation by the exchange of the integration data structure references.
2)   We evaluate the performance of our proposed strategy through industry standard benchmarks and demonstrate that our strategy achieves significant performance improvements.

The paper is organized as follows: Section 2 first gives a motivating example. Section 3 introduces the integration data structure in details. A simplified application scenario is examined to evaluate the design. Section 4 introduces the by reference data transfer in details. Section 5 is experiment and discussion. Section 6 introduces related works. Section 7 sums up with several concluding remarks.

2   Motivating Scenario

In this section, we describe a motivating scenario to explain the problem to be addressed by the reported research.
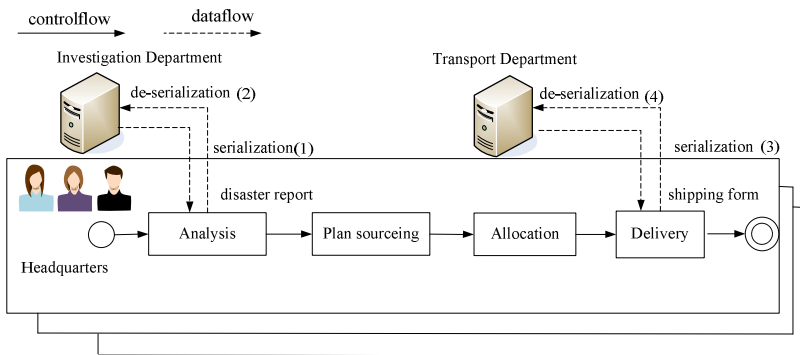
Fig. 1. The SOAP data transfer

This scenario describes a part of the Red Cross headquarters' relief management process after a disaster as shown in Fig. 1. The Red Cross headquarters has to coordinate among several affiliated organizations to supply the required materials for a relief effort as soon as possible. Firstly, an officer collects and analyzes the disaster report to make plan. To support the disaster report, several kinds of data are required. Then he notifies the transport office to deliver the materials. Since all tasks are implemented through Web services, so the process executor has to invoke the Web services in each process instance. If there are a large number of concurrent instances, we can anticipate a very high cost of data transfer.

3   An Integration Data Structure

Let's still consider the motivation example. Disaster report generally describes many factors which have great influences on the materials supplies, such as the location, weather, victims, and loss. This paper only considers three factors: disaster victims, disaster area, and disaster loss. To invoke the analysis Web service, the officer has to extract data, and transform them into the format in accordance with the complex data type described in Web Service Description Language (WSDL). Fig. 2(a) shows the

complex data type named "DisasterReport", but the process of "DisasterReport" serialization and de-serialization is a time-consuming process.

In order to support the officer to define the equivalent data structure while reducing the cost of serialization and de-serialization, this paper uses a data structure based on nested relational model, namely nested table. The reasons that the nested table is used as the fundamental data structure are that any complex object can be described by nested relation according to the theory of nested relations and complex objects in databases [Abiteboul, S. 1989]. At the top of Fig. 2(b) shows the nested relational model corresponding to the "DisasterReport" complex data type, namely data schema. The "DisasterReport" relation specifies a list of disaster locations in certain area. It has one atomic attributes("address") and three sub-relations named "DisasterVictim", "DisasterArea","DisasterLoss". Each atomic attribute is a primitive data type in the format of "-type", where type includes int, string and double. The sub-relations describe the casualties, areas and losses at each disaster location.

In the nested table, data are represented as nested table instance. At the bottom of Fig. 2(b) shows an example of nested table instance, which describes the disaster victim, disaster area and disaster loss at each disaster location. A nested table uses the data service to present the underlying data. The data sources are the information resources in the format of HTML, XML, CSV or Relational Table. They become data services through data binding as shown in Fig. 2(c), and the binding result must accord with the data schema. At runtime, the data services are instantiated to generate nested tables, which are saved in key-value database. The reason that we adopt the key-value database is that traditional relational database has disadvantages of scalability and query complexity as a result of strict Normal Form theory.

(a)complex data type   (b) nested table   (c)data binding
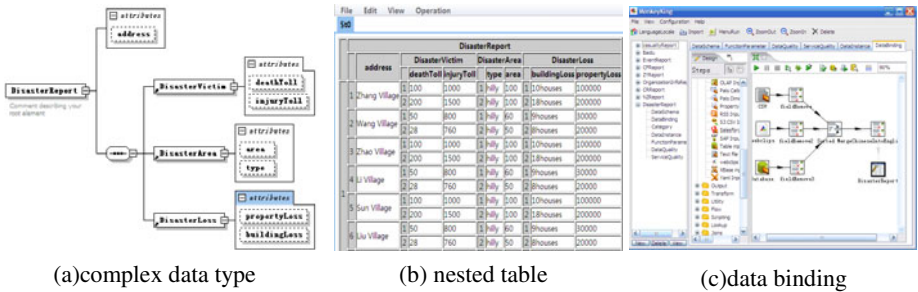
Fig. 2. The integration data structure

After that, the parameter value can be directly obtained as long as the data type described in WSDL is the same as the data schema. The nested table can be considered as an intermediate data structure from the source data to the Web service, and carries the data aspects for the functional Web service. In order to guarantee data availability, the system regularly incrementally updates the nested tables through the comparison between last data binding execution and the new data binding execution. After the nested table generation, the officer can refer to these nested tables as the input of business service [Han, Y.B 2009].

Aside from the transmission of regular data, such as file address or manual input string, on the client side, the officer selects the nested table "DisasterReport" as input. In fact, the selected result is only the reference, such as the nested table reference "nested table: 4d63b7e2dad994288f70a9f4", and then the information is included in the SOAP message for the invocation of analysis Web service.

4   By Reference Data Transfer

Before introducing the section, we have to claim a prerequisite that each server publishing Web service mush install our syntax analyzer and type transformer, where the syntax analyzer is responsible for connecting to the key-value databases in cluster, and enables server to obtain the nested tables according to their references, and the type transformer is responsible for obtaining the parameter value of Web service after receiving the nested tables from the syntax analyzer. Moreover, there is a default local file to cache the nested tables at each server. The entire "by reference" data transfer is shown in Fig. 3(a).

(a) "by reference" data transfer   (b) analyzing and transforming (c) the synchronization process
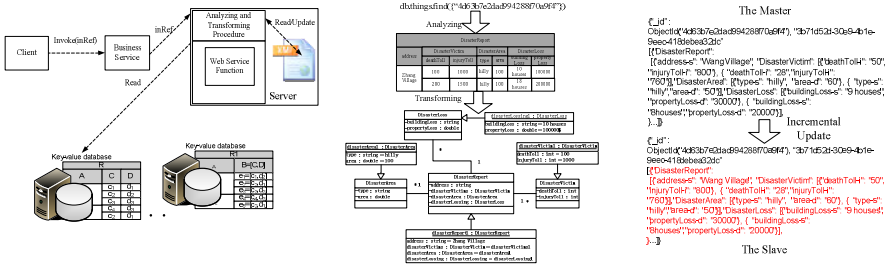
Fig. 3. The "by reference" data transfer

Compared with the SOAP data transfer, the "by reference" data transfer has two advantages. The first advantage is that the SOAP serialization and de-serialization within the Web service invocation are replaced with the less cost syntax analysis and type transformation as a result of HTTP transport. The second advantage is that most of the steps of the syntax analysis and type transformation are conducted incrementally on the local servers.

Let's introduce an example to show the feasibility of "by reference" data transfer. In the example, we use the MongoDB[1] as the key-value database. Through syntax analyzing, the syntax analyzer finds the nested table reference "nested table: 4d63b7e2dad994288f70a9f4" in SOAP message, where "nested table:" indicates the nested table reference, and the "4d63b7e2dad994288f70a9f4" is identical to the

---

[1] http://www.mongodb.org/

identifier of the "DisasterReport" nested table in the key-value database. The syntax analyzer receives the nested table through HTTP and sends it to the type transformer. The type transformer generates four JAVA classes named "DisasterReport", "DisasterVictim", "DisasterArea" and "DisasterLoss" corresponding to the data schema, and generates a JAVA object named "disasterReport1" coressponding to the nested table instance, where "disasterReport1" represents the first instance of "DisasterReport", "1" represents the sequence number of instances, as shown in Fig. 3(b). The type transformer sends the JAVA object to the analysis program, and then the analysis program is executed and the analysis result is returned. We use rsync master slave mode[Tridgell, A. Mackerras, P. 1996] to synchronize the nested table, where the "DisasterReport" in cluster is written into a master directory and the "DisasterReport" in server is written into a slave directory, and they are synchronized through rsync algorithm, as shown in Fig. 3(c).

5   Experiment and Discussion

The experiment is to compare "by reference" data transfer with SOAP data transfer to get the conclusion that "by reference" data transfer can adapt to highly concurrent environment. We test their performance respectively through industry standard benchmarks from WSTest[2]. WSTest includes the following benchmarks:

1) EchoStruct - receives an array of arbitrary length as input parameter and returns this array. The structures in the array contain one element each of type integer, float, and string. The longer the array, the more work is required in de-serialization and re-serialization of the SOAP object to and from XML.
2) EchoList - sends and receives a linked list of any size, where each element in the list consists of the same structure as used in EchoStruct.

Firstly, we use the EchoList benchmark to build 20 similar Web services with different number of columns as input and transmit the "DisasterReport" respectively, where each column includes string values of length 500. For each Web service, we build a nested table to represent the corresponding input. Fig. 4(a) shows the transmission time regarding different number of columns for the SOAP and the nested table approach. As we can see, the nested table approach outperforms the SOAP approach. The SOAP message structure, gets more complex, the performance of the SOAP transmission gradually gets worse compared to large messages with simple structures. Moreover, Fig. 4(b) illustrates that the SOAP transfer, including SOAP serialization and de-serialization at the client side and the server side, makes up almost 80% of the whole SOAP approach, and the fact proves that the process of SOAP serialization and de-serialization is a time-consuming process from performance perspective, and our "by reference" data transfer is a good choice. Secondly, we continue to use EchoList benchmark. In like manner, we need to build a nested table as shown in Fig. 2(c). The

---

[2]http://msdn.microsoft.com/en-us/netframework/cc302396

client initiates concurrent threads to invoke the Web service with structured message, which is an array consisting of the "DisasterReport". The number of invocations executed and the response time are accumulated during a period of 500 seconds and are reported at the end of the execution.

As shown in Fig. 4(c) and Fig. 4(d), we note that the performance improvement of the nested table over SOAP increases with the number of client threads. This suggests that the "by reference" data transfer is a scalable solution that can adapt to highly concurrent environment.

| (a) | (b) | (c) | (d) |
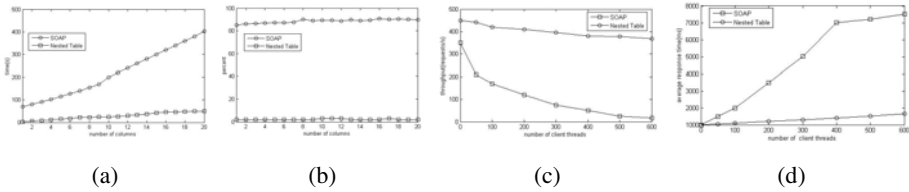
Fig. 4. The "by reference" data transfer evaluation

6   Related Works

The performance of the SOAP protocol has often been regarded relatively poor due to its use of XML encoded messages. Existing optimization technologies mainly are categorized into implementation technologies and modeling technologies.

On implementation technologies: In paper [Cai, M., Ghandeharizadeh, S. et al 2002], the authors present a compression technique that employs XML semantic to reduce network bandwidth. In paper [Nayef A. G., Michael J. L. 2005], the authors present a new optimization technique called differential de-serialization (DDS), which exploits the similarities between incoming messages to reduce de-serialization time. In paper [Li, W.B., Tordsson, J. et al 2010], Li et al. propose an approach to consistency-preserving caching and compression of Web service response messages to minimize data transfers, but the approach only works efficiently on condition that there is a number of similar results.

On modeling technologies: Heinzl et al present a novel and more flexible way of handling attachments in SOAP-based Web service environments [Habich, D., Preissler, S. et al 2007]. In paper [Habich, D., Richly, S. et al 2008], the authors propose a specific extensions on the Web service as well as the BPEL levels. Using the extended technologies, the exchange of large data sets between participating services in a process is conducted with specific data-oriented approaches, such as ETL tools for the data exchange between databases. But the approach has two disadvantages as follows: 1) the specific data transfer services have to be available at process execution time; 2) the data transfer is still a time-consuming activity in the face of concurrent invocations. On the contrary, our proposed integration data structure supports the business user programming, and adapts to high concurrent invocations according to the experiments.

7   Conclusion

In this paper, we propose a dataflow optimization for service-oriented application. To be more comprehensive, we will adapt our dataflow optimization to more general service composition language such as WS-BPEL.

References

1. Abiteboul, S.: Nested relations and complex objects in databases. Springer (1989)
2. Cai, M., Ghandeharizadeh, S., Schmidt, R.R., Song, S.: A Comparison of Alternative Encoding Mechanisms for Web Services. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453, p. 93. Springer, Heidelberg (2002)
3. Habich, D., Preissler, S., Lehner, W., Richly, S., Amann, U., Grasselt, M., Maier, A.: Data-grey-box web services in data-centric environments. In: The International Conference on Web Services (2007)
4. Habich, D., Richly, S., Preissler, S., Grasselt, M., Lehner, W., Maier, A.: BPEL-DT - data-aware extension of BPEL to support data-intensive service applications. In: Emerging Web Services Technology, vol. II (2008)
5. Han, Y.B., Wang, J., Zhang, P.: Business-oriented service modeling: A case study. Simulation Modelling Practice and Theory 17(8), 1413–1429 (2009)
6. Li, W.B., Tordsson, J., Elmroth, E.: An Aspect-Oriented Approach to Consistency-Preserving Caching and Compression of Web Service Response Messages. In: 2010 IEEE International Conference on Web Services, pp. 526–533 (2010)
7. Ghazaleh, N.A., Lewis, M.J.: Dierential deserialization for optimized soap performance. In: Proceedings of the ACM/IEEE SC 2005 Conference on High Performance Networking and Computing (2005)
8. Tridgell, A., Mackerras, P.: The rsync algorithm. Tech. Rep. TR-CS-96-05, Canberra 0200 ACT, Australia (1996), http://samba.anu.edu.au/rsync/

Group-Scope Query and Its Access Method

Yi Wang, Fan Xia, and Aoying Zhou

Software Engineering Institute, East China Normal University, China
{yiwang,fanxia}@ecnu.cn, ayzhou@sei.ecnu.edu.cn

Abstract. Nowadays, large scale workloads of typical Web 2.0 applications are well supported by NoSQL systems. The most efficient query that NoSQL systems have provided us is the key-lookup, but it is not quite efficient for some kinds of complex queries. One of these kinds of queries, which we name as Group-Scope Query, is very common in practical applications. In this paper, we explore an access method for group-scope queries. In this method, data are partitioned into groups, and a multi-dimensional primary index is built for each group. Meanwhile, we avoid establishing secondary indexes which are known to require additional expenses. YCSB is used to evaluate the performance of our approach, and the results show that our access method is scalable and efficient.

1 Introduction

With the rapid development of Web 2.0 applications, thousands or millions of users may simultaneously login their websites and do writes, updates and reads. These operations are relatively simple, which may only involve key-lookups, reads and writes of one record or a small number of records, but the scalability is highly required. Currently, NoSql(Not Only Relational or Not Relational) systems support the typical workload of web 2.0 applications by providing good horizontal scalability.

However, there always exists the requirement for more complex queries. Rather than all kinds of queries, what we focus on here are named as "group-scope query". For instance, in the application of short message system, a user want to find all the messages that satisfying some conditions, and there may exist thousands of messages under a user. This kind of queries are common, with the feature that their conditions always contain a certain attribute being equated to a constant. This attribute is not a primary key, such as userId in the previous example. The selection scope can be narrowed to a relative small group by using the first-attribute as a filter. NoSql systems employ global secondary indexes or a map/reduce mechanism for advanced queries[1], which will introduce long latency and lose the simple and fast read-write-style that NoSql systems have delivered to us. It is overkill to use global secondary indexes or a map/reduce mechanism for group-scope queries.

Our approach is to group and partition all the data by a first-attribute, and build a multi-dimensional primary index for each group. This paper makes the following contributions:

1. Tuples, which belong to different relations, with the same first-attribute value are clustered, and these tuples have high probability to be access together; Multidimensional hashing using Gray Code is employed to cluster consecutive and similar records, which will decrease the number of random access reading(up to 50% in theory[2]);
2. Multi-dimensional primary indexes are used in our approach, which are less expensive than secondary indexes and provide the flexibility that secondary indexes provide. Suppose a multidimensional index is built on an attribute set A, then this index can substitute any indexes built on subset of A in theory. Moreover, computational overhead has been greatly reduced after optimization.
3. Experimental evaluation by YCSB[3] shows the scalability and efficiency of our approach.

2   Related Work

In current NoSQL movement, encouraged by the great success of Google's BigTable[4] and Amazon's Dynamo[5], a number of NoSQL systems emerge and present us with simplicity, flexibility and scalability. Based on their different data models, NoSql systems are classified into three categories[1]: key-value stores, document-style stores and extensible record stores. The basic data model of extensible record stores, analogous to Bigtable, is rows and columns, which is more complex than the key-value model in key-value stores. Compared with document stores, the extensible stores don't have a flexible data model (the column value doesn't support nested objects), but they are designed for higher throughput. Although secondary indexes are available for advanced selection, queries based on indexed attributes are much less efficient than key-lookups and key range scans. Typical systems include BigTable, HBase, HyperTable and Cassandra[8].

3   Problem Statement

In this paper, we arrange data in NoSql systems and provide an efficient access method for group-scope queries. Here we give the definition of "group-scope query".

Definition 1 (group-scope query). A group-scope query *is a query, whose logical query plan has a form of* $\sigma_{A=c\ AND\ C}(R)$ *(selection σ is a basic operator in relational algebra, R is a relation, A is an attribute, c is a constant and C is some other condition).*

Usually, in a lookup operation, A is a primary key and the query result will be one tuple. As for a group-scope query, A, named as "first-attribute", isn't a primary key. In typical web 2.0 applications, a first-attribute could be the personID of personal profiles, the e-mail account of e-mails, the phone number of short messages, the userID of online dating records, the keyword of classified ads, etc. After tuples are filtered by the condition about the first-attribute, there could still exist hundreds or thousands of tuples that need to be iterated and filtered by C. Condition C may contain multiple attributes.

4   Data Arrangement

Relations having the same first-attribute are saved in a sorted map indexed by a key which is the value of the first-attribute. A number of tuples, which have the same value of the first-attribute, are stored together as the "value" corresponding to a "key" in the map. Each key-value pair is a row, so the key is also known as the row-key.

4.1   Storage Management

We store data in extensible record stores. Rows of a sorted map are sharded to nodes by row-keys. Different extensible record stores provide different sharding strategies: simple hashing, key range assignment. Hash indexes, ordered hash indexes or B-tree indexes built on the row-key are used for searching a particular key or key range over nodes. Most extensible record stores cache their writes and updates in memory, and data are sorted. These sorted memory buffers will be flushed to disk when reaching a certain size or periodically. File compaction happened periodically. Those data files are not dynamically organized.

A group-scope query $\sigma_{A=c \text{ AND } C}(R)$ can be rewrite to $\sigma_C(\sigma_{A=c}(R))$. Locations of the row that contains the tuples satisfying the condition $A = c$ can be found through using a row-key index. As there are many tuples under a row, we would rather use a index under this row than scan all the tuples and filter for what belong to relation R and satisfy condition C. We choose to build a multi-dimensional primary index for each row on a subset of all the attributes under these relations. We choose Gray ordering as the multidimensional access method. A bitwise interlaced key is generated for each tuple, called as fingerprint. Tuples under each row are sorted by fingerprints and stored as sequential files. A one-dimensional index will be built for each row. B+ tree could be used for particular big row data; otherwise a sparse index fitting in a few blocks can just meet the requirement. We call these indexes as fingerprint indexes.

4.2   Construction of Fingerprint

Fingerprint is a n-bit Gray code, with a c-bit prefix that is unique among different relations. In order to get the content part of the fingerprint for a tuple, we firstly use an order preserving hashing function to map each value of indexed attributes into a Gray code. Supposing $A_1, A_2, \ldots, A_d$ are d indexed attributes under a relation, a n_i-bit Gray code represents a value of the attribute A_i, and $\sum_{i=1}^d n_i = n - c$. h_{A_i} is the order preserving hashing function for A_i, $\{v_i\}_{i=1}^d$ are the values on the indexed attributes of a tuple t under the relation R and let $B2C(.)$ be the transfer function from nature binary codes to Gray codes. Then, $h(v_i)$ will map v_i to a integer, and $\{B2C(h(v_i))\}_{i=1}^d$ will be the result of the first phase. Secondly, we construct the fingerprint, a n-bit Gray code as a whole, by interleaving these bit strings of d attributes in an order defined by a shuffling

function. The shuffling function is an array of size n, where the elements are chosen from 1 to d. The fingerprint K for t:

$$K = \Psi(t) = \mathrm{prefix}(R) + \mathrm{shuffle}(f, \{B2C(h(v_i))\}_{i=1}^{d}),$$

where $\mathrm{prefix}(R)$ is the prefix part of R, and f is the shuffling function for R. For example, $d = 2, n_1 = 2, n_2 = 2$, and $f = \{2, 1, 2, 1\}$; therefore, $\mathrm{shuffle}(f, \{"00", "11"\}) = "1010"$. Tuples in a row are ordered by their fingerprints according to the principles of Gray code.

5   Query Processing

We establish a SQL query interface above extensible record stores. After query compilation and generation of the logical query plan tree, the leave operand has a form of $\sigma_C(\sigma_{A=c}(R))$. As mentioned in Sect. 4.1, a row-key index is used to access the row in which tuples satisfy the condition $A = c$. If fingerprints of R are generated based on attributes $\{A_1, A_2, \ldots, A_d\}$, then the condition C can be expressed as $C(A_1, A_2, \ldots, A_d)$ (the condition that only involves the attributes in $\{A_1, A_2, \ldots, A_d\}$) AND D(for some other condition). The select operator $\sigma_C(\ )$ may be implemented as

(a) Converting $\sigma_{C(A_1,A_2,\ldots,A_d)}$ to a one-dimensional query in the fingerprint space,
(b) Fetching tuples via fingerprint indexes, and
(c) Filtering tuples by condition D.

The rest of Sect. 5 will focus on Step (a) – conversion. $\sigma_{C(A_1,A_2,\ldots,A_d)}$ can be regarded as a multi-dimensional query on d attributes. Conversion needs to handle two different situations: The exact match on d attributes can easily perform with good performance; for the partial match and the multi-attribute range query, mapping from multi-dimensional constraints to one-dimensional range constraints is much more difficult. Unlike the general space-filling curves approach[6], our new algorithm takes advantage of the principles of Gray code to solve this problem, which dynamically processes queries based on the distribution of data. It consists of three parts:

1. (Sect. 5.1)the translation part is to get a set of wildcarded fingerprints. A wildcarded fingerprint is a string of n characters from the set $\{"0", "1", "?"\}$, where "?" stands for "don't care", and it can represent fingerprints having the same form;
2. (Sect. 5.2)the transformation part is to get non-overlapping sorted-intervals which will dynamically adjust the interval size according to the distribution of data;
3. the last part is to do one-dimensional range queries, and the specific approach is decided based on the type of the index built for fingerprints(B+ trees or sparse indexes).

5.1   Translation

We will firstly prove that any region query could be translated to a set of wild-carded fingerprints. Then, we will introduce our approach for translation of range queries.

Theorem 1. *All the possible fingerprints for a certain region query can be arranged into the largest possible groups containing $2^m(m = 0, 1, 2, \ldots, n)$ adjacent codes. Every group can be described as a certain string (wildcarded fingerprint) containing "1", "0" and "?". For instance, "1?0?" represent a group of size $2^2\{"1000", "1001", "1100", "1101"\}$.*

Proof. We define the domains of all the attributes as $\{D_1, D_2, \ldots, D_d\}$. q is supposed to be a region query with the search region $\Omega \subseteq E^d = D_1 \times D_2 \times \cdots \times D_d$. Let S be all the possible fingerprints that the returned tuples of q could have. S can be regarded as a set of true values of a certain boolean function. With the aid of Karnaugh map[7], which is a method to simplify Boolean algebra expressions, the boolean function can be simplified as the smallest sum-of-products function. The minterm, generated through the axiom laws of boolean algebra, can be represented as a wildcarded fingerprint. All the wildcarded fingerprints represent the groups that we want S to be divided into.   □

Algorithm 1. find the minterm set M_i for interval $[l_i, u_i]$ in domain D_i

---

Input: l_i, the lower bound of the interval; u_i, the upper bound of the interval; n_i, the
  number of bits representing the value in D_i.
Output: M_i, the minterm set for the interval;
1: **if** $[l_i, u_i]$ is a trivial interval **then**
2:      $M_i = \{"?^{n_i}"\}$ or $M_i = \{h_i(l_i)$ or $h_i(u_i)\}$
3: **else**
4:      $l = h_i(l_i)$, $u = h_i(u_i)$
5:      $M_i = \text{DQ}(l, u, M_i)$
6:      $M_i = \text{Kmap}(M_i)$ ▷ the procedure of the Karnaugh-map algorithm
7: **end if**
8: **return** M_i
9:
10: **procedure** $\text{DQ}(l, u, M)$ ▷ a recursive function
11:     **if** $l \leq u$ **then**
12:         $(i, n) = \max_i\{(i, n)| l \leq n2^i, (n + 1)2^i - 1 \leq u\}$
13:         $M = M \cup \{B2C(n) + "?^i"\}$ ▷ "+" is a operator to concatenate two strings
14:         $M = \text{DQ}(l, n2^i - 1, M)$
15:         $M = \text{DQ}((n + 1)2^i, u, M)$
16:     **end if**
17:     **return** M
18: **end procedure**

As for a range query, Ω is a d-dimensional interval $I^d = [l_1, u_1] \times [l_2, u_2] \times \cdots \times [l_d, u_d]$. In theory, it's possible to find minterms of all the possible fingerprints

for a range query. However, if just Karnaugh map is used, the processing will have high overhead. As a result, an optimized approach is applied to solve this problem.

Property 1. Suppose M_i is the minterm set of the range query with the search interval $[l_i, u_i]$ in the domain D_i ($i = 1, \ldots, d$), then $\text{prefix}(S) + \text{shuffle}(f, M_1 \times M_2 \times \cdots \times M_d)$ is the minterm set of the range query with the search range $I^d (= [l_1, u_1] \times [l_2, u_2] \times \cdots \times [l_d, u_d])$.

Prop.1 indicates that we can firstly calculate M_i in each domain, then get all the minterms through product and shuffling. Algorithm 1 shows how to get the minterms for interval $[l_i, u_i]$. The optimized algorithm is much more efficient than the naive approach simply using Kmap. Suppose there is a range constraint $[l_i, u_i]$ on A_i and the difference between $h_i(l_i)$ and $h_i(u_i)$ is N, the cost is decreased from $O(N^2)$ to $O(\log^2(N))$. Referring to Algorithm 1 line 5, the terms that need to be processed through Kmap are limited to $O(\log(N))$ with the help of a divide-and-conquer procedure. N will be extremely large if a relatively many bits used to represent a domain or a relatively large range to query for; however, $\log(N)$ will not be larger than n_i.

5.2   Transformation

Transformation, illustrated in Algorithm 2, is to transform the wildcarded fingerprints(minterms), gained from previous translation processing, to non-overlapping intervals.

In order to provide a dynamical optimized transformation for data with different distributions, a preprocessing for minterms is performed before transformation. Suppose $p(x)$ is the probability density function of fingerprints in a data file or a node, where x varies from 0 to $2^n - 1$. Moreover, C is the number of tuples in the data file or in the node and a is the number of tuples that a block can hold. Then, $F(y, \delta) = \int_y^{y+\delta} \frac{C}{a} p(x) \mathrm{d}x$ means the number of blocks needed to store the tuples whose fingerprint values are in $[y, y + \delta]$. Our purpose is to find the proper size of the interval so that the qualifying tuples can fill a block. y is a random variable, and let $F(y, \delta) = 1$. Then, interval size δ is a function of y. The expectation of δ, $\mathrm{E}(\delta)$, can be estimated. The overall formula is

$$\mathrm{E}(\delta) = \oint_{\int_y^{y+\delta} \frac{C}{a} p(x) \mathrm{d}x = 1} \delta \mathrm{d}y.$$

Let $l = \log_2 \mathrm{E}(\delta)$. Rightmost l bits of fingerprints which are the inputs of Algorithm 2 are replaced with "?". Then the outputs of Algorithm 2, the transformed intervals, will have suitable size. Each data file and each node will maintain their own l(hereinafter referred to as l_f and l_n, respectively). When a node processes a query, it will use its own l_n by default. If the difference between l_f of the data file to be accessed and defaulted l_n is greater than a certain threshold, l_f instead of l_n will be used.

Algorithm 2. transform a term to intervals

Input: a term $m = "[01?]^{n-n_t-1}[01]?^{n_t}";$ ▷ Exponentiation of a character implies
    repetition, and square brackets imply arbitrary choice of the enclosed characters
Output: R, the interval set;
 1: truncate the tail of m
 2: $R = \{("","")\}$
 3: **repeat**
 4:     $c \leftarrow$ the last character of m, $m \leftarrow m$ truncate c
 5:     **if** c is "0" or "1" **then**   ▷ "+" is a operator to concatenate two strings
 6:         each $(r_l, r_u) \in R$ is replaced by $(c + r_l, c + r_u)$
 7:     **else**
 8:         populate each $(r_l, r_u) \in R$ with $("0" + r_l, "0" + r_u)$ and $("1" + r_l, "1" + r_u)$
 9:         if r_l has pattern $"1[0]^p$, merge two intervals to $("0" + r_l, "1" + r_u)$
10:     **end if**
11: **until** the length of m equals to 0
12: adjust each $(r_l, r_u) \in R$ to be a interval, and enlarge the interval by 2^{n_t} times
13: **return** R

---

6   Evaluation

Our approach could be implemented in all extensible record stores, and we choose
Cassandra as the base system which is a derivative of Amazon's Dynamo and
currently draws lots of attentions of commercial application areas. We use the
latest stable version Cassandra 0.8 up to the time this paper is written, and
plugin a SQL query interface to define relation schemas as well as to parse and
execute queries.

We evaluate the performance of our approach using the Yahoo Cloud Serv-
ing Benchmark (YCSB) framework(Version 2.0), which defined a core set of
benchmarks to evaluate cloud systems and is extensible to add new workloads.
Our workload includes write operations and group-scope queries. The number
of tuples under each row follows a uniform distribution. The data type of the
attribute is number or string, and the value of the attribute follows either a
normal distribution or a Zipfian distribution.

Our testing infrastructure includes 6 cluster nodes which are connected to-
gether to simulate a cloud computing platform. Communication bandwidth is
1Gps. Each node has a 2GHz Intel Core8 Quad CPU and a 40G disk. The op-
erating system is Ubuntu 9.10 Server OS. In order to evaluate the performance
of accessing data in disks, memory amount allocated to JVM is limited to 2G,
and row cache of Cassandra is set to 0.

Fig. 1 (a) illustrates performance of group-scope queries. Our versions get
higher throughput and lower latency than the original version (Cassandra). Af-
ter using the optimization method mentioned in Sect. 5.2, we get even better
performance. Fig. 1 (b) shows that our version doesn't introduce any additional
overhead for the performance of write operations.

(a) query operations

(b) write operations
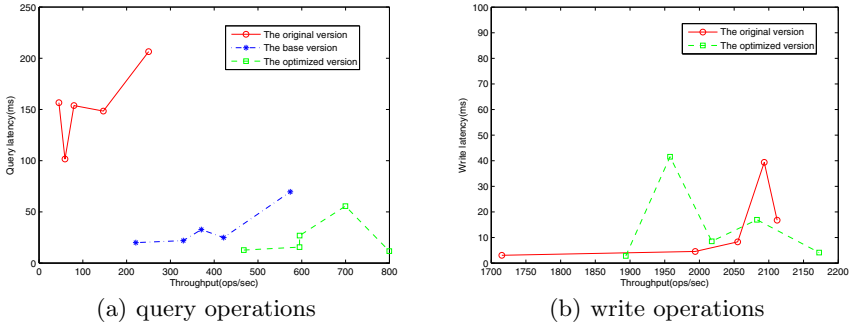
Fig. 1. The original version is Cassandra 0.8, in the base version we have applied our approach without query optimization mentioned in Sect. 5.2, and the optimized version includes optimization

7   Conclusion

In this paper, an access method is explored for group-scope queries which are very common in large scale workloads of web 2.0 applications. The feature of group-scope queries is that one of the conditions is a certain attribute (called as first-attribute) being equated to a constant. Our access method is to arrange tuples with same first-attribute value in a same row and build a multidimensional primary index for each row. Multidimensional hashing of Gray Code is used to make consecutive and similar tuples clustered, which will decrease I/O cost. Computational overhead has been greatly reduced after optimization, and the algorithm of query is dynamically adjusted according to the distribution of tuples. The experiments verify that our approach is scalable and efficient.

References

1. Cattell, R.: Scalable sql and nosql data stores. ACM SIGMOD Record 39(4), 12–27 (2011)
2. Faloutsos, C.: Multiattribute hashing using gray codes. In: ACM SIGMOD Record, vol. 15, pp. 227–238. ACM (1986)
3. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with ycsb. In: Proceedings of the 1st ACM Symposium on Cloud Computing, pp. 143–154. ACM (2010)
4. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS) 26(2), 1–26 (2008)
5. Hastorun, D., Jampani, M., Kakulapati, G., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W.: Dynamo: amazons highly available key-value store. In: Proc. SOSP. Citeseer (2007)
6. Tropf, H., Herzog, H.: Multidimensional range search in dynamically balanced trees. Angewandte Info., 71–77 (1981)
7. Karnaugh, M.: The map method for synthesis of combinational logic circuits. Trans. AIEE. pt. I 72(9), 593–599 (1953)
8. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review 44(2), 35–40 (2010)

Introducing SaaS Capabilities to Existing Web-Based Applications Automatically[*]

Jie Song, Zhenxing Yan, Feng Han, Yubin Bao, and Zhiliang Zhu

College of Software, Northeastern University, Shenyang, P.R. China
songjie@mail.neu.edu.cn, yanzhenxing1234@126.com,
hf81970@163.com, {baoby,zzl}@mail.neu.edu.cn

Abstract. In recent years, SaaS has been widely used in web-based applications due to its special characteristics. It brought both users and software providers' benefits and profits. It is significative if service providers can automatically convert existing web-based applications to SaaS paradigm (named as *SaaSify*). So an effective and automatically *SaaSify* approach is needed urgently. In this paper, we analyze and conclude the new challenges of *SaaSifying* web applications, design a series of *SaaSified Command* for introducing SaaS capabilities to existing web-based applications automatically; and propose *SaaSified Command Set* and rules to adapt the variety of both original and *SaaSified* applications. Finally, we describe a case study to show the effectiveness of proposed approach, the performance experiments prove that the proposed approach is effective and efficient.

Keywords: Cloud Computing, Software as a Service (SaaS), *SaaSify*.

1 Introduction

Cloud computing has become a very hot term in the last two years, due to its appearance as an effective reuse paradigm[1], where software functionality, hardware computing power, and other computing resources are delivered in the form of service so that they become available widely to consumers. SaaS (Software as a Service) as a model of cloud services, delivers the conventional software functionality as a service, it is more popular in small and medium-size business because of its flexibility, high customization, and comfortable price strategy.

   Most SaaS applications provide services through web. However, large quantities of existing web-based applications have been built in non-SaaS ways. Service providers will reduce the cost greatly if they can convert existing applications to SaaS paradigm (named as *SaaSify*) instead of redeveloping them. Unfortunately, *SaaSifying* a huge web-based application manually is still costly both in time and resource [2]. **Firstly**, the engineers would take more time to understand the business logics of origin application and convert them to SaaS paradigm. **Secondly**, they have to be familiar

---

with the architecture of both the origin application and the *SaaSified* one, and figure out how to change the former to the later. **Thirdly**, for properly *SaaSifying*, other aspects such as database and runtime environment should also be well considered. Compared with developing a new application,SaaSifying the existing one will reduce the workload by 40% [3]. Therefore, *SaaSify* approach which can automatically convert a non-SaaS application into a SaaS one is in urgent need.

Since it is difficult to *SaaSify* an application manually, it would be even more difficult to design an automatically *SaaSify* approach. According to open-closed principle, software entities should be open for extension, but closed for modification [4]. Therefore, a basic challenge the proposed approach faces is how to introduce SaaS abilities to original application without modifying its source codes. In that case, the automatically *SaaSify* is feasible. The detailed challenges are explained as following:

- **Variety of sources:** The *SaaSify* approach should adapt various features of the origin application, such as programming language, architecture and runtime environment.
- **Extension but not modification:** It's difficult to introduce SaaS characteristics to the original application without modifying the source codes and do that automatically.
- **Variety of targets:** The approach should adapt the different SaaS maturity level.

To solve the challenges mentioned above, we propose a *SaaSify* approach which can convert web-based applications to SaaS paradigm automatically. We propose *SaaSified Rules* to encapsulate various features of the original applications; design *SaaSified Command*s to introduce SaaS characteristics to original applications; organize the *SaaSified Command*s into *Command Sets* to meet each SaaS maturity level; Experiments and case study prove that the proposed approach can solve the challenges well, it is effective and efficient.

2   Related Works

The SaaS related researches cover many aspects, and among which, approaches of modeling and building SaaS application are the most concerned topic. Few researches focus on *SaaSifying* automatically so far. We give some relevant works as follows.

[2] pointed out the advantages and difficulties of converting traditional systems to SaaS platform, it exactly means *SaaSifying*. The authors surveyed several cases of SaaS service, and identified the common key functions of SaaS service as well as two important axes of maturity model. Their point of view on *SaaSifying* coincides with ours. However, [2] only proposed the solution to build practical SaaS service without further discussing automatic *SaaSifying*.

The following papers give samples of SaaS implementation or manually *SaaSify* existing web applications. [5] improved traditional medical simulation, education and evaluation system to support SaaS. [6] introduced a online-payment platform based on SaaS. [7] introduced a *SaaSified* teaching resources management platform. It seems that people start to focus on *SaaSifying*, so our research is proved to be meaningful.

[8] is the most related work, it provided an end-to-end methodology and toolkit for fine granularity SaaS-ization. SaaS-ization means *SaaSifying* in practice. The toolkit

really benefits the developers when they are *SaaSifying* applications. However, that proposed toolkit helps developers to SaaS-ization applications, the developers have to modify the source code as well as the server configuration. Our approach is focus on the binary applications (almost), it is automatic, unnecessary and unable to change the source codes. Even so the experimental approaches of [8] still contribute to our work greatly.

Generally, proposed *SaaSify* approaches belongs to software reengineering, automation and modernization, it is a new topic brought by era of cloud computing. There are several works related to *SaaSifying* methodology and experience. However, these works are not automatic, convenient and general solutions. Compared to the others, our *SaaSify* tool is proved to be more meaningful, useful and innovative.

3   SaaSified Command

As we know, SaaS applications have some characteristics such as configurable, multi-tenant, and scalable. *SaaSifying* is to introduce these characteristics to original applications. Based on these characteristics, we define a series of *SaaSified Command*s in this section.

3.1   Basic *SaaSified* Command

A *SaaSified Command* encapsulates a single process for introducing a corresponding SaaS characteristic or the smallest functional point to an original web-based application. In this section, we will define and list some critical *SaaSified Commands.*

Definition 1 Operator: Operator is defined as an atomic operation on files. It has three modes which are "*copy*", "*update*" and "*delete*". They are basic operation in a command.

Noticing that "*update*" is allowable on file, but it does not means modifying source code, only file written by descriptor language or plain text, such as HTML page and configuration file, or tables in database, can be modified.

Definition 2 Command: Command represents a logical operation for introducing a corresponding SaaS characteristic or the smallest functional point to original web based application, it is defined as a 3-tuple:

$$Command=<N_c, T_c, I_c>$$

Among them: N_c represents the name of *command*; T_c is the target file processed by the *command*; I_c represents the concrete implementation of the *command*, for example, a class in a object-oriented language. We list the designed *command*s by their N_cs as following:

- F_C (**Format Command**), which formats (*updates*) the files, such as HTML and JSP files in view layer. It is designed for the requirements of configuration.
- DC_C (**Data Configure Command**), which replaces (*updates*) the database connector of the original application, is designed for supporting multi-tenant and data isolation.

- WC_C (**Web Configure Command**), which *updates* the web configuration, add listeners and filters, is designed for intercepting tenants and context information.
- DU_C (**Database Update Command**), which *updates* database such as creating tables, altering column and building new reference, etc. is designed for storing tenant information in the original database.
- RL_C (**Runtime Library Command**), which *copies* the runtime library to the original application, the runtime library is dependent library developed by us for supporting SaaS abilities.
- LU_C (**Login Update Command**), which *updates* the login page, add the tenant-infor related page elements, is designed for providing login interface of both tenants and users.
- FC_C (**Function Configure Command**), which extracts (*copies*) functions points of original application and assembles them as a configuration file, is designed for supporting "configurable functionalities" of SaaS.
- PC_C (**Page Configure Command**), which divides (*updates*) web pages into many blocks, and assigned these blocks to tenants, is designed for supporting "configurable interfaces" of SaaS.
- B_C (**Backup Command**), which *copies* a file to the backup location, is designed for backup the file before it is processed.
- R_C (**Recovery Command**), which *copies* a file from the backup location the application, is designed for recovering the file when exceptions happen.
- D_C (**Delete Command**), which *deletes* files, is designed for erasing the intermediate files during the *SaaSify* process.

All the *SaaSified Command*s are designed for supporting the characteristics of SaaS paradigm. They focus on how to add the SaaS characteristics to the original application. These *command*s can resolve the problem of converting a non-SaaS application to a SaaS one automatically. *SaaSified Command*s listed above are sketchy, and some critical ones will be described in the next section.

3.2   Critical *SaaSified* Command

In this section, we will explain the detailed mechanism of some *SaaSified Command*s, which are critical ones to our *SaaSifiy* approach, the rest of others are abbreviated.

1)   Page and Function Configure Command (PC_C & FC_C)

There are two kinds of configurabilities in SaaS paradigm, the first one is "configurable interfaces", and the other one is "configurable functions". We can not know functions of the original application because it is blank-box to proposed approach, so we extracting the function list by analyzing buttons and links in pages. Above all, web pages should display according to tenants configurations. **Firstly**, the position and color of page elements should be accord with tenants' configurations; secondly, tenants can only see the links and buttons whose corresponding functions have been paid. PC_C is adopted to support the former, named "configurable interfaces" and FC_C is adopted to support the later, named "configurable functions".

To implement "configurable interfaces", we divide the original page into blocks, then it is possible to select and regroup this blocks by configurations. The PC_C follows seven steps:

Step 1: Scan the HTML sources of pages

Step 2: Divide the page into visual blocks according to the visual effect as shown in Fig. 1, visual effect may presents by fixed tags such as *<td>*, *<tr>* and *<p>*.

Step 3: Analyze the HTML code of each visual block, extract them into page blocks;

Step 4: Number these blocks, re-implement page blocks, distinguish them from each other by adding some HTML invisible tags such as *<div>*.

Step 5: Assign the page blocks to tenants' configuration.

Step 6: Implement algorithm for freely compositing page blocks into page by script language.

Step 7: When a user visits a page, the page generator retrieves the tenant-info which the user belongs to, and dynamically regroup the pages according to tenant's configuration.
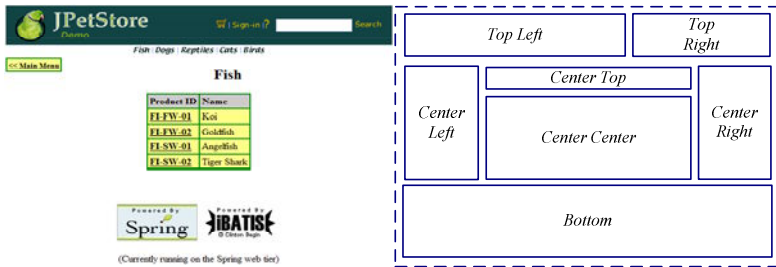
Fig. 1. Example of Dividing page blocks

To implement "configurable functions", we extract URL from links and buttons in each page block. Then it is possible to show or hide these elements selected by configurations. The FC_C follows seven steps:

Step 1: Scan all links and buttons of each page blocks, extracting their URLs.

Step 2: Analyze this URLs, retrieve a proper name from each URL as the function name. In practice, these names are not intuitive, therefore, we provide an interface to let the administrator replace these namespace with intuitive ones. Some non-functional URLs are also be excluded by administrator.

Step 3: Make these links and buttons independent from pages by adding some HTML invisible tags such as *<div>*.

Step 4: Assign the functions to tenants' configuration.

Step 5: Implement algorithm for freely show or hide these links and buttons by script language.

Step 6: When a user visits a page, the page generator retrieves the tenant-info which the user belongs to, and dynamically show the links and buttons whose corresponding functions have be paid by current tenant.

Step 7: Introduce a request filter to the web server, for example, add a filter servlet to "*web.xml*" in JavaEE application, make sure even the unpaid URLs are requested, they will be discarded.

2) Runtime Library Command (RL_C)

This RL_C is designed for importing the runtime libraries to original application. The runtime library is dependent library developed by us for supporting SaaS abilities.

Most of the web-based applications adopt B/S (Browser/ Server) architecture, and they are divided into three layers: view layer, business logic layer and data access layer. The critical function of runtime libraries is maintaining the tenant-info, making it available for all three layer. Traditionally, user-info is persisted in database and then loaded to the session as soon as a user logs on. It is passed to business logic layer and data access layer where session is unavailable. In a *SaaSified* web application, both user-info and tenant-info should be managed. However, tenant-info cannot be stored in session, because we cannot change the source codes to let tenant-info pass to business logic layer and data access layer.

Since a user-thread is available to every layer during user's once request, we consider loading tenant-info to a memory block in user-thread. A filter is added to handle the request, filter the logon information and find out tenant identification, then load tenant-info to current thread. Thus, logic and data layer can share tenant-info.

3.3   Command Set

According to [9], SaaS architectures generally can be classified as belongings to one of four "maturity levels", whose key attributes are of configurability, multi-tenant efficiency, and scalability. Each level is distinguished from the previous one by the addition of one of those three attributes. There are *Ad-Hoc/Custom* (level 1), *Configurable* (level 2), *Multi-Tenant-Efficient* (level 3) and *Scalable* (level 4). When we *SaaSify* a web application, the *SaaSify* approach should adapt to each maturity levels. The architecture of SaaS maturity level 1 has no difference with the traditional application, and that of level 4 is mainly implemented by servers distribution (hardware), so that when *SaaSify* a web application, only level 2 and 3 are well considered.

We classified two *Command Sets*, each set includes the different versions of *SaaSified Command*s which are designed for adapting two SaaS maturity levels. Administrators can choose the right *Command Set* according to the chosen SaaS maturity level. Two *Command Sets* and their including *command*s are listed as follows:

- *Command Set* for Level 1 and 2: includes B_C, R_C, D_C, F_C, FC_C and PC_C;
- *Command Set* for Level 3 and 4: includes B_C, R_C, D_C, F_C, FC_C, PC_C, DC_C, WC_C, DU_C, RL_C and LU_C.

The *SaaSified Command Set*s proposed in this paper can resolve the problem of variety maturity levels during *SaaSifying*.

4   SaaSified Rule

The *SaaSified Command*s are designed for introducing SaaS characteristics to non-SaaS web applications. However, the original applications have various features, such as coding language (main focus in this paper), runtime environment, architecture etc. A fixed *command* couldn't adapt all these features. So the *command*s are divided as two different types: *Universal Command* and *Replaceable Command*. They can be described as follow:

- **Universal Command:** *Universal Command* can adapt all types of original applications, it is independent from features of original applications, it includes B_C, R_C, and D_C.
- **Replaceable Command:** *Replaceable Commands* adapt the diverse features of original applications. Clients can choose the suitable *Replaceable Commands* to execute the *SaaSify* process according to the coding languages of original applications. *Replaceable Commands* include F_C, DC_C, WC_C, DU_C, RL_C, LU_C, FC_C and PC_C.

A *Replaceable Command* has some substitutes. This can be understood as the relationship between abstraction and its concretion. We propose *SaaSified Rule* to define the *Replaceable Command* and its concretions. The definition is given as follow:

Definition 3 *SaaSified* Rule: Let a *Replaceable Command rc* has N concretions $\{ rc_i' \mid 0 < i < N \}$, *condition$_i$* is the mapping conditions from *rc* to *rc$_i'$* , A *SaaSify* rule R_{rc} is defined as:

$$R(rc)=\{<\textbf{If } condition_i \textbf{ Then } rc_i'> \mid 1 \leq i \leq N \}$$

condition$_i$ stand for the various feature of original application, Given the *JSP*, *ASP* and *PHP* as three languages for implementing dynamic web pages, the follows list *SaaSified* rules for introducing the "page configurability":

R(PageConfigureCommand):
If *JSP pages* **Then** *JspPageConfigureCommand*;
If *ASP pages* **Then** *AspPageConfigureCommand*;
If *PHP pages* **Then** *PhpPageConfigureCommand*.

The *SaaSified* rules are stored in the external rule repository which is represented by XML file. The Fig. 2 gives a fragment of "*SaaSifiedRule.xml*". In the rule file, every "*SaaSifiedRule*" item represents a rule and has three "*Maps*". Every "*Map*" includes two attributes which are "*condition*" and "*action*", respectively representing coding language (such as ASP, PHP and JSP) and the concrete executive *command*.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <SaaSifiedRules>
  - <SaaSifiedRule id="FC">
      <Map condition="jsp" action="cn.edu.neu.SaaSify.JspFormatCmd" />
      <Map condition="asp" action="cn.edu.neu.SaaSify.AspFormatCmd" />
      <Map condition="php" action="cn.edu.neu.SaaSify.PhpFormatCmd" />
    </SaaSifiedRule>
  + <SaaSifiedRule id="DCC">
  + <SaaSifiedRule id="WCC">
  + <SaaSifiedRule id="DUC">
  + <SaaSifiedRule id="RLC">
  + <SaaSifiedRule id="LUC">
  + <SaaSifiedRule id="FCC">
  - <SaaSifiedRule id="PCC">
      <Map condition="jsp" action="cn.edu.neu.SaaSify.JspPageConfigureCmd" />
      <Map condition="asp" action="cn.edu.neu.SaaSify.AspPageConfigureCmd" />
      <Map condition="php" action="cn.edu.neu.SaaSify.PhpPageConfigureCmd" />
    </SaaSifiedRule>
</SaaSifiedRules>
```

Fig. 2. Fragment of SaaSifiedRule.xml

SaaSified Rule file is loaded to memory during *SaaSifying*. Given a special *Replaceable Command*, its concrete *command* is found and executed. The *SaaSified* rules proposed in this section can effectively resolve the variety of original application, and *Replaceable Command* also has expansibility of unforeseen changes.

5   Evaluations

We have developed the *SaaSify* tool based on proposed *SaaSify* approach. In this section, we will evaluate the proposed *SaaSify* approach by case study and performance analyzing.

5.1   Cases

The Java Pet Store is a well-known standard application given by Sun Co.. It is designed to illustrate how the Java Enterprise Edition 5 Platform can be used. It is convinced if our *SaaSify* tool can *SaaSify* Pet Store. Some related works such as [8] also chose Pet Store as an example case to prove the correctness and applicability of their methodology.

Based on the rule in section 3, the tenants are able to configure their own system and manage their own data after the configuration data are saved in database. The Fig. 3 show the differences between the pages which belong to tenants *A*(left figure) and *B*(right figure) with distinct configuration and data.

Fig. 3. Page of Fish Category in Java Pet Store for Tenant *A* and Tenant *B*

In Fig. 3, red rectangles are marked to highlight the differences.

- **Page style:** The position and color of the navigation, button and list are different (see area 1-3).
- **Functions:** Tenant *B* has a search bar at the top of the page while tenant *A* does not (see area 3).
- **Queried data:** The information displayed in the tenant *B*'s data list is different compared   with that of tenant *A* (see area 4).

5.2   Performance

We design series of experiments to compare the performances of original "Java Pet Store" and the *SaaSified* one. We focus on three measures to evaluate the performance; they are Transaction Response Time (*TRT*), Processor Time (*PT*) and

Memory Usage (*MU*). In the interest of space we do not present the experimental data which could be found in our another work [10]. And we know that:

- *TRT* of the *SaaSified* is a little higher than that of the *Original*, the additional cost on maintaining the tenant-info is worth to trade-off of SaaS functionality.
- *PT* and the *MU* of the *SaaSified* are apparently lower than those of the *Original*, because of the advantage of *SaaSified* architecture.

Above all experiments, the *SaaSify* tool has passed the planned test cases, correctly converted Java Pet Store to a SaaS application, and the *SaaSified* applications shows the performance advantages in experiments. Since Java Pet Store is widely considered as a standard JavaEE application, we can draw a conclusion by analogy that it can also convert other JavaEE applications to SaaS applications. Therefore, the *SaaSify* approach can be proved reasonable and effective.

6   Conclusions

As SaaS model is widely accepted, *SaaSify* tool which can automatically convert a web-based application into a SaaS application is significative. In this paper, and holistic *SaaSified* solutions is proposed and some detailed mechanisms are explained. The primary work of this research can be divided into the following four aspects:

- Analyze and conclude three challenges of automatically *SaaSify* web-based application;
- Design a series of *SaaSified Command*s to introduce SaaS *abilities* to oriented application, and explain the details of the critical ones. The *SaaSified Command*s successfully solve the challenge of introducing SaaS characteristics to the original application without modifying its source codes. And *Command Sets* are proposed to adapt the different SaaS maturity levels;
- Design the *SaaSified* rules and make process easy on the diverse features of original applications;
- Plan the performance experiments to prove the proposed approach efficient and effective.

All of these approaches and implantations are contributed to spreading SaaS technology, increasing usability, and enhancing the applicability of existing web-based applications. This work is further research of [10]. Future work of this research includes building the *SaaSify* tool based on this *SaaSify* approach in a more user-friendly way, for example as an eclipse plugins.

References

1. La, H.J., Kim, S.D.: A Systematic Process for Developing High Quality SaaS Cloud Services. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) CloudCom 2009. LNCS, vol. 5931, pp. 278–289. Springer, Heidelberg (2009)
2. Kang, S., Myung, J., Yeon, J., Ha, S.-w., Cho, T., Chung, J.-m., Lee, S.-g.: A General Maturity Model and Reference Architecture for SaaS Service. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010, Part II. LNCS, vol. 5982, pp. 337–346. Springer, Heidelberg (2010)

3. Wang, Y., Zhang, B., Liu, Y., Wang, D.: The Modeling Tool of SaaS Software. In: International Conference on Advanced Computer Control (ICACC 2010), pp. 298–302. IEEE Press, Shenyang (2010)
4. Meye, B.: Object Oriented Software Construction, California (1988)
5. Shang, J., Yang, J.J., Wang, Q., Pan, H.: The Framework of Medical Simulation Education and Evaluation System for Supporting SaaS. In: 34th Annual IEEE Computer Software and Applications Conference Workshops (COMPSACW), Seoul, pp. 93–97 (2010)
6. Liu, L.R., Song, M.N., Luo, X.X., Bai, H.P., Wang, S.B., Song, J.D.: An implementation of the online-payment platform based on SaaS. In: SWS, Beijing, pp. 658–662 (2010)
7. Zheng, J.C., Li, X.Q., Zhao, X.X., Zhang, X.M., Hu, S.: The research and application of a SaaS-based teaching resources management platform. In: ICCSE, Hefei, pp. 765–770 (2010)
8. Cai, H., Zhang, K., Zhou, M.J., Gong, W., Cai, J.J., Mao, X.S.: An End-to-End Methodology and Toolkit for Fine Granularity SaaS-ization. In: 2009 IEEE International Conference on Cloud Computing, Bangalore, pp. 101–108 (2009)
9. SaaS Maturity Model,
   `http://www.saas-attack.com/saasmaturitymodel.aspx`
10. Song, J., Han, F., Yan, Z.X., Liu, G.Q., Zhu, Z.L.: A SaaSify Tool for Converting Traditional Web-Based Applications to SaaS Application. In: Proc. 2011 IEEE International Conference on Cloud Computing, pp. 396–403. IEEE Press, Washington DC (2011)

Computing Resource Prediction for MapReduce Applications Using Decision Tree

Jing Tai Piao and Jun Yan

School of Information Systems and Technology
University of Wollongong
Northfields Avenue, Wollongong, NSW, Australia
{jp928,jyan}@uow.edu.au

Abstract. The cloud computing paradigm offer users access to computing resource in a pay-as-you-go manner. However, to both cloud computing vendors and users, it is a challenge to predict how much resource is needed to run an application in a cloud at a required level of quality. This research focuses on developing a model to predict the computing resource consumption of MapReduce applications in the cloud computing environment. Based on the Classified and Regression Tree (CART), the proposed approach derives knowledge of the relationship among the application features, quality of service, and amount of computing resource, from a small training. The experiments show that the prediction accuracy is as high as 80%. This research can potentially benefit both the cloud vendors and users through improving resource management and reducing costs.

Keywords: Cloud computing, Decision tree, Machine learning, MapReduce, Resource Prediction.

1 Introduction

Cloud computing has unveiled the new era of large-scale and data-intensive computing. By means of virtualizing underlying physical computation resource, a computer cluster could be formed in a more economical and flexible way so that the computational resource could be requested. The data parallel applications which were previously exclusive for high performance computing are able to be executed on the virtualized cluster rather than expensive physical computers. As a parallel programming model, MapReduce has been implemented by major cloud computing service providers, such as Amazon Elastic MapReduce [13]. MapReduce simplifies the parallel application development with Map function and Reduce function regardless of task synchronization [6, 13]. Under such a circumstance, huge amount of CPU cores and memory would be assigned to tremendous virtual machines to support MapReduce applications. Management of computing resource is a challenge to both cloud users and cloud computing providers. On the one hand, cloud users have problems estimating the computing resource requirements to run their applications at a required level of quality. For instance, the customers of Amazon Elastic MapReduce

often select the number and type of instances either arbitrarily or based on a rough estimation. As a consequence, the customers may unnecessarily pay for resource that is not needed for running the application, or the execution of the application is not compliant with the required quality level, e.g., the execution takes longer time due the shortage of resource. On the other hand, from the service providers' perspective, they have difficulties to improve the efficiency of the resource utilization by always allocating adequate amount of resource to applications.

Obviously, it is desirable that the computational resource requirements of an application can be predicted before the application is actual run. It will potentially benefit both cloud users and cloud vendors. Cloud users would be able to request computational resource from the cloud in a more economical way. At the same time, cloud vendors would be able to schedule their resource in a more efficient manner.

The rest of the paper is structured as follow. The next section reviews the major related research. Then, Section 3 gives an overview of the MapReduce applications, followed by an introduction of the CART algorithm in Section 4. After that, Section 5 discusses the proposed prediction model in details. Section 6 presents the experimental results and compares the prediction accuracy of our approach with K-nearest neighbour algorithm. Finally, Section 7 concludes this paper and outlines our future work.

2   Related Work

Machine learning approaches have been adopted by a number of researchers to predict the application computing resource consumption, in terms of CPU number, memory and disk size, and so on. By means of collecting historical application execution data as the training set, the prediction problem could be modelled as supervised machine learning problem [1, 5]. Fairly intensive work has been done on this problem and the major research contribution is reviewed in the following.

Linear regression assumes that the relationship between dependent variable (y) and independent variables (x) can be described as $y = \beta x + \varepsilon$. Theoretically, it can be expected that the completion time of an application would decrease as the computing resource being consumed increases. This approach is applied in [1] to simplify the sophisticated modelling problem for medical image process applications. In practice, however, the increased computation resource may not be related to the decline of completion time in a linear way [13]. Rather, a numbers of approaches have proved that predicting more accurately than linear model [3, 4, 12].

Support vector machine (SVM) has advantages in terms of handling a large number of attributes in the non-linear scenario, whereas the disadvantage of SVM is that it brings extra computational overheads at the same time and it relies heavily on the size of training set [2].

In the [3], the characteristics of application are formed in multiple dimensions in the space. Thus, the Euclidean distance between the applications in the space could describe the similarity among these applications. The advantage of this kind of

approaches is that the prediction result could be highly accurate. But the accuracy usually relies on the abundant historical data and the computational complexity would be relatively higher than other approaches.

Decision tree algorithms ignore the correlation between attributes, classifying the training set according to its characteristics. On each node of the tree structure, a separate criterion will be used to divide the training set into different categories. The separate process would not stop until the purity of each leaf on the tree is satisfied. C4.5 decision tree algorithm is implemented in [10] to classify the training set into different time intervals. However, the time intervals are static so that the control granularity cannot be optimized. In [5, 8, 9, 11, 12], the application attributes are selected and grouped as subsets so that the searching can be done in the templates instead of the tree structure. The characteristics of applications are described as a template, such as several predefined attributes, and thereby the similarity between applications could be judged by comparing each attribute. Further, the research results have proven that the prediction accuracy of template approach is better than adopting complex regression as characteristics split method [12].

3   Overview

This research is based on the primitive assumption that applying the same computing resource to similar applications would have similar performance. Also, this research is inspired by the success of prediction of application runtime according to the pre-execution features in previous research [4, 7, 10, 12]. It is possible that predicting an application's computing resource demand by analysis data from previous similar application.

The CART algorithm provides such a way to classify the applications by splitting characteristics of application one by one. It has a significant advantage in reduced size of training set, comparing with other machine learning algorithms such as SVM and K-nearest neighbour algorithm [9].

In order to apply CART to predict computing resource requirement, the historical execution information of applications, including application features, performance and computing resource consumption should be collected as the training set. The data of computing resource are chosen as dependent value, whereas other variables are chosen as independent variables. We assume that the computing resource can be combined by several fixed templates. This is similar to the way that, Amazon EC2 offers several types of instances to build a cluster with arbitrary computational capacity. Thus, the computing resource is predefined as classes, whereas the application features are presented by multiple attributes.

In this model, the performance of application is represented by execution time, while the application features are defined as several attributes. The CART can be applied on the data by setting the variables of performance and application features as predictors.

In CART, the size of the tree determines the accuracy of prediction significantly. However, as the tree is constructed deeper and deeper, the computation complexity

would increase simultaneously. In order to balance the accuracy and computation efficiency, we implement CART by the following steps:

1. Construct the tree with a minimum size,
2. Observing the prediction accuracy p_i, and
3. if $p_i < 80\%$, reconstruct a deeper tree i_k, until $p_i \geq 80\%$

In the tree construction phase, the original learning set is split according to the impurity function. For each node of CART, the impurity function has a maximum value if all classes are mixed equally and has a minimum value when the node contains only one class. Among all of the possible splitting, the best way should offer the largest decrease in impurity.

4 The Prediction Approach

4.1 Prediction Model

The object of this research is to predict the computational resource demand of a MapReduce application. Most of the previous solutions request collecting and accumulating a huge amount of application historical execution data. Our decision tree based approach could not only provide contented prediction accuracy but also dramatically reduce the size of training data set. The decision tree approach, besides, reduce the computational complexity, comparing with other distance based algorithms so that the learning process could be much faster than other approaches.

The core problem of our research is how to model the application features so that the training set is not tedious but effective for prediction. In our approach, we describe each application using a 6-tuple:

{TaskCompletionTime, dataSize, dataDistributionNumber, dataSiteNumber, MapperNumber, ReducerNumber}

- TaskCompletionTime means the time period from the first mapper starts to the last reducer completes.
- dataSize is the total size of the workload. The unit of this attribute is Megabyte (MB) in this approach.
- dataDistributionNumber means the number of workload distribution. In the case that the workload of the application is stored separately and performance of the application might be influenced by the number of data pieces, the data distribution number is introduced to describe the number of pieces the workload has been divided into.
- dataSiteNumber shows the number of sites which are used to save the workload. Mapper and reducer numbers are straightforward values to show how many mappers and reducers are setup in the application.

The historical application execution data will be collected according the above model. By mean of implementing decision tree algorithm, the similar application can be classified into same leaf on the tree. Then, a split rule can be extracted from this tree.

For a new coming application, it is possible to classify the application belongs to which leaf by implementing the rule on it.

In this research, each leaf on the tree indicates a computer resource class. In practice, Amazon Elastic MapReduce has classified its computational resource as different classes, such as standard instances, high memory instances and high CPU instances, and so on. Each type of instance has different computational capacity. For example, a default small instance of Amazon Elastic MapReduce runs a 32-bit platform with 1.7 GB of memory, 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit), 160 GB of instance storage, whereas a High-Memory Double Extra Large Instance runs a 64-bit platform with 34.2 GB of memory, 13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each), 850 GB of instance storage [14]. In our case, we form our experimental computing resource as four classes which is similar to the instance classification of Amazon Elastic MapReduce. The computation resource categories include:

- Class 1: highest performance type which has 2 CPU core (3.33 GHz) and 2GB memory,
- Class 2: high performance type which has 1 CPU core (3.33 GHz) and 1 GB memory
- Class 3: standard type with 1 CPU core (3.33 GHz) and 512 MB memory,
- Class 4: economic type with 1 CPU core (2.40 GHz) and 512 MB memory

The following table shows a part of the training sample:

Table 1. A part of the training samples

Time(second)	Data Size(Mb)	Data Number	Site Number	Mapper	Reducer	Classes
1700	4.4	1120	1	1	1	2
1530	2.6	994	1	1	1	2
1583	2.8	1036	1	1	1	2
1726	2.9	1094	1	1	1	2
1490	2.5	976	1	1	1	2
1728	3	1130	1	1	1	2
1735	2.9	1121	1	1	1	2
2326	69.1	1515	1	1	1	2

In this research, the application is specified as classic word count application and the work load is randomly generated with size ranging from 69MB to 0.7MB which consists of 3028 to 226 files, respectively. The word count application was executed under four computational environments as predefined. The training set is collected from the 200 observations which consist of 50 samples for each class.

After the accumulation of the training data, the CART algorithm will be adopted to construct the tree. Fig. 1 shows a part of the tree which contains 15 nodes and the accuracy reaches 53.7%.
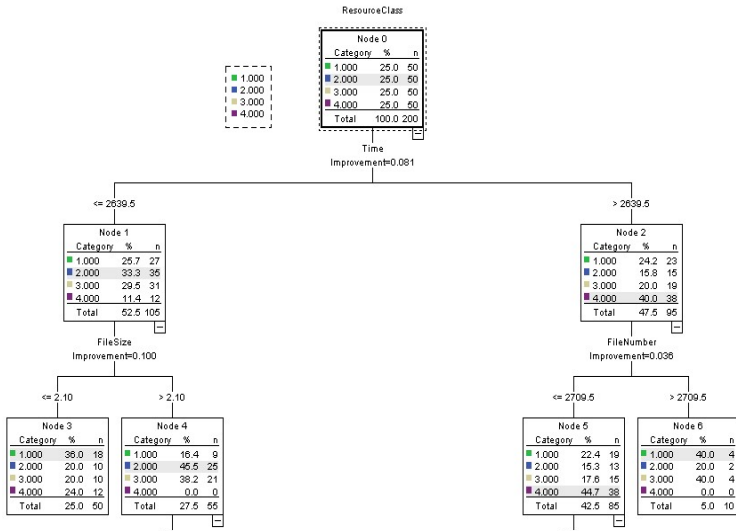
Fig. 1. A part of the CART in this research

5   Evaluation

This research conducts N-fold cross-validation method which randomly divides the sample set as several subsets ($X_1, X_2, ..., X_n$) to evaluate the prediction accuracy because the size of training sample. For each subset n, using $X - X_n$ as learning sample and n as test sample, the accuracy estimation will be the average of n test results.

Fig.2. shows accuracy estimation result as the number of nodes increases. Here, the accuracy is estimated by n–fold cross validation with n = 10. The prediction accuracy increases as the tree constructed deeper. While the prediction accuracy achieves the predefined value (80%), the construction of the tree will be terminated.
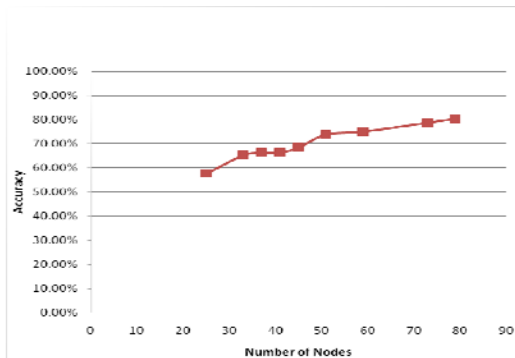
Fig. 2. The prediction accuracy of CART

In the next phase, k-nearest neighbour algorithm is implemented to the same learning set. The parameter of k ranges from 1 to 10. Also, the accuracy is measured by n-fold cross validation and n =10. The prediction result can be seen in Fig. 3.
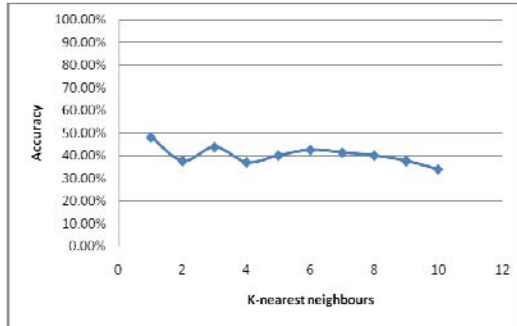
Fig. 3. The prediction accuracy trend of k-nearest neighbor algorithm

According to the experiment result, we found that CART can achieve higher prediction accuracy by using a small training sample k-nearest neighbour algorithm which only has accuracy from 34.1% to 48.2%.

6   Conclusion

We have demonstrated a CART based method to predict the computing resource consumption of a MapReduce application within the context of cloud computing. The success of previous research which estimated the performance of applications before actual execute it proves the feasibility of predicting the computing resource demand. The prediction of computing resource demand would have significant values in improving current cloud computing service quality, such as Amazon Elastic MapReduce. By means of such prediction mechanism, the user could request adequate computing resources from the cloud so that the application can be executed in a more economic way.

Compared with previous predicting algorithms, CART takes advantages of a small training sample set so that the predicting result could be obtained earlier than the other approaches. According to the experimental results, we found that the prediction accuracy of our approach is much higher than k-nearest neighbor algorithm which is another famous machine learning algorithm.

In the future, a client side application would be developed to improve the current service of Amazon Elastic Cloud Computing based on the proposed CART algorithm. The application would allow the users to specify expected execution time before submitting MapReduce applications. To do so, the application features in the prediction model will be extended to a general model.

References

1. Albers, R., Suijs, E., de With, P.H.N.: Triple-C: Resource Usage Prediction for Semi-Automatic Parallelization of Groups of Dynamic Image-Processing Tasks. In: Proc. of the 23rd Int. Parallel Distributed Processing Symp. (2009)
2. Duan, R., Nadeem, F., Wang, J.: A Hybrid Intelligent Method for Performance Modeling and Prediction of Workflow Activities in Grids. In: Proc. of the 9th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), Shanghai, China, pp. 339–347 (May 2009)
3. Ganapathi, A., Chen, Y., Fox, A.: Statistics-Driven Workload Modeling for the Cloud. In: ICDE Workshops 2010, pp. 87–92 (2010)
4. Ganapathi, A., Kuno, H., Dayal, U., et al.: Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning. In: Proc. of the 2009 IEEE International Conference on Data Engineering, Shanghai, China, pp. 592–603 (March 2009)
5. Gibbons, R.: A Historical Application Profiler for Use by Parallel Schedulers. In: Feitelson, D.G., Rudolph, L. (eds.) IPPS-WS 1997 and JSSPP 1997. LNCS, vol. 1291, pp. 58–77. Springer, Heidelberg (1997)
6. Kaashoek, F., Morris, R., Mao, Y.: Optimizing MapReduce for Multicore Architectures, technical report,
   `http://dspace.mit.edu/bitstream/handle/1721.1/54692/`
   `MIT-CSAIL-TR-2010-020.pdf?sequence=1`
7. Matsunaga, A., Fortes, J.: On the Use of Machine Learning to Predict the Time and Resources Consumed by Applications. In: Proc. of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), Melbourne Australia, pp. 495–504 (June 2010)
8. Mu'alem, A.W., Feitelson, D.G.: Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling. IEEE Transactions on Parallel and Distributed Systems 12(6) (June 2001)
9. Mitchell, T.M.: Machine Learning, McGraw-Hill Science/Engineering/Math (March 1, 1997)
10. Nadeem, F., Fahringer, T.: Using Templates to Predict Execution Time of Scientific Workflow Applications in the Grid. In: Proc. of the 9th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), Shanghai, China, pp. 316–323 (May 2009)
11. Guim, F., Rodero, I., Corbalan, J., et al.: The Grid Backfilling: a Multi-Site Scheduling Architecture with Data Mining Prediction Techniques. In: CoreGrid Workshop in Grid Middleware (2007)
12. Smith, W.: Prediction Services for Distributed Computing. In: Proc. of IEEE Internatioal Parallel and Distributed Processing Symposium, Long Beach, US, pp. 1–10 (June 2007)
13. Smith, W., Foster, I., Taylor, V.: Predicting Application Run Times Using Historical Information. In: Feitelson, D.G., Rudolph, L. (eds.) JSSPP 1998. LNCS, vol. 1459, pp. 122–142. Springer, Heidelberg (1998)
14. Zaharia, M., Konwinski, A., Joseph, A.D., Katz, R., Stoica, I.: Improving MapReduce performance in heterogeneous environment. In: OSDI 2008 Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (2008)
15. `http://aws.amazon.com/elasticmapreduce/`
16. `http://hadoop.apache.org`

Who Resemble You Better, Your Friends or Co-visited Users[*]

Jinjing Ma and Yan Zhang[**]

Department of Machine Intelligence, Peking University, Beijing 100871, China
Jinjing.ma@pku.edu.cn, zhy@cis.pku.edu.cn

Abstract. People are alike their graphic neighbors in social networks, which is generally accepted as the basic assumption in interest prediction. But what kind of neighbors is a better information source? This paper aims at answering this question by comparing the results of predicting users' interests in blog social networks with different relationships and parameters. Since social networks usually keep "friends" and "visitors" as basic social roles, we take these two online social relationships as the main information source. In this paper, we discover that (1) combining different information sources might lead to better prediction, and (2) there are many other factors that can affect the results significantly.

Keywords: social network, interest prediction.

1 Introduction

Interest prediction is widely used in recommendation systems. For example, based on the social correlations among its users, *flickr* provides rankings of contents [3]. And online stores like *dangdang.com* apply collaborative filtering systems to recommend products. In fact, there are two assumptions widely used in recommendation systems.

Assumption 1. People share similar interests with their neighbors in social networks;
Assumption 2. Pages clicked by users with similar interests or relation networks tend to share similar contents.

As we can see, these two methodologies are functional in the researches of user interest prediction and applications [1, 2]. Then what will happen if we apply them on the same platform and make a combination? This paper aims at answering this question.

From *Assumption 2*, we have a deduction.

Deduction 1. Users visited by the same user probably share similar interests.
Based on *Deduction 1* we introduce the virtual relation among those visited by the same users (defined as co-visited users).

To answer the question we put forward at the beginning, we take friendship, 2-degree friendship, and co-visited relation into account. Here, we do prediction with friendship based on *Assumption 1*, with co-visited relation based on *Deduction 1*, and with 2-degree friendship as extension. With which we conclude that the larger number of one's co-visited users gathered information from, the higher possibility we have to infer his overall perspective of interests. We want to make a step in discovering if different relations have different utilities in interest prediction.

In this paper, we choose a blog site, since there are many social networks built with the basic function of a blog site, like posting photos and videos can be regarded as transformed blogs since they are all pointing to expressing. Thus, though SNSs have different information flows when focusing more on user interactions, understanding the structure of a blog site might help us to do more complex researches on social networks like *Facebook*. For example, sharing or forwarding friends' contents can be seen as the combination of accepting and publicizing information. In addition, the relationships on a blog site are clean since there are few relations except friendship. Thus, we can exclude some disturbances in advance.

Within this context we make the following contributions:

- We compare predictions based on different relation networks separately, and we discover that their performances are quite different. When considering different linear combined results of them, we find that some of the combinations lead to improvement but others do not.
- We import some other parameters to modify the results of prediction. And we evaluate their effects on predictions. Thus, we discover that some of the parameters can affect the results to a large extent.

This paper is organized as follows. In Section 2 we review related work to prepare the necessary background for user modeling and prediction method. The methodology including the prediction model and the evaluation principle is presented in Section 3. The way we establish the dataset and comparison experiments performed are discussed in Section 4. In Section 5 we analyze the results of the experiments and conclude with comments and future plans of our work.

2   Related Work

Our work, which is focusing on the comparison of predicting results of users' interests with various relationships, is related to prior approaches to interest modeling, social tags as well as friendship in social networks.

In social networks, users may express or hint their interests in various ways. A commonly used source is user-published content, including articles [10], and author-generated tags [5, 6]. And a research classifies the tags into topics and finds that online users use more details to describe the contents they prefer [9]. When comparing different sources for interest prediction, we take advantage of the simplicity of user-generated tags.

Modeling social correlation and network structure is a fundamental work for interest prediction, and has been extensively studied by researchers, including social scientists [11]. When users grow more similar due to social influence and those with

similar interests tend to develop new relationships in the future, a research finds clear feedback effects between the two factors, thus *Assumption 1* is supported [7].

There are mainly two trends of predicting interests with neighbors in online network: visitors and friends. For example, collaborative filtering system is applied in interest prediction [8]. Similarly, a research combines interests of those who have visited the same pages [12]. Another example is Wen and Lin's work, which starts to take social cues from two-degree friends into consideration [4]. And the latest research employs a co-clustering method to a discriminative learning method to predict users' preference [13].

3   Methodology

3.1   User Interest Model

Like many other works, we define users' interest as bags of words. Here we consider tags generated by the authors as the best summarization of their topics, which are provided by our data-source blog site. Thus the result is strongly related to users' subjective judgments. In addition, we merely include tags from people's ten latest blogs within 3 months, thus these bags of words can stand for their short-term interests on the time point just before crawling, and the time series produced by long-term interest drift is not within the scope of the study.

Particularly, we attach an attribute called "preference" to those tags. For a user *u*, the set of tags which *u* likes to refer to recently is defined as *tag(u)*, and his preference of a chosen tag *w* is

$$pref(w|u) = \begin{cases} 1, w \in tag(u) \\ 0, w \notin tag(u) \end{cases}. \tag{1}$$

3.2   Visit-Relationship and Co-visited Relationship

Visit-Relationship. On social networks, "visit" is the most common behavior among users. Even the person might be interested in the content or title of the pages while visiting, except for randomly clicking, it is unfair to equate the contents people like to visit with those they like to express. Therefore we think it inappropriate to treat users' visitors' interests as source for prediction.

Co-visited Relationship. As we have discussed above, people can be interested in the pages they visit when they are not randomly scanning. Based on *Deduction 1*, we treat users' co-visitors (users both visited by the same user) as sources to predict their interests. In addition, the blog site on which we perform our experiments provides latest 20 visitors of a user, which is the source to build up the co-visited network.

3.3   Friendship and 2-Degree Friendship

Friendship. On most blog sites, "friends" and "visitors" are clearly defined. When two users accept each other as "friend" online, they build up "friendship". Obviously, a friend's visiting can be regarded as part of friendship. In this paper, we define friendship as the relationship between two friends and visit-relationship as relationship between users and their visitors, who are not friends. In this way, this can be only one of the two links between two bloggers, if they have a link.

2-Degree Friendship. 2-degree friendship is a bidirectional link between a pair of users having common friends. And 1-degree visit-relation is unidirectional; co-visited relation is a two-way link. In the experiment, we separately produce relationship networks F_1 by friendship, F_2 by 2-degree friendship, V_1 by visit-relationship, and V_2 by co-visited relationship.

3.4   Strength of Relationship

In this paper, "strength" functions as a weighting factor in the predicting model. In general, a relation's strength is related with the frequency of interaction, as well as the similarity of social circles and interests. Here the strength of relation $\overrightarrow{u_i u_j}$ can be regarded as how much u_i influences the prediction of u_j's. In addition, it is noteworthy that for a bidirectional relation $\overrightarrow{u_i u_j}$, there can be one strength per direction. Moreover, we apply two kinds of simplified strengths in our experiments, which we will introduce in the following. u_i and $u_j (i \neq j)$ are users in the relationship graph G ($G \in \{F_1, F_2, V_2\}$).

The first definition assumes that initial strength of relation $\overrightarrow{u_i u_j}$ is

$$str_1(u_i|u_j, G) = \begin{cases} 1, & \exists \overrightarrow{u_i u_j} \in E(G) \\ 0, & \nexists \overrightarrow{u_i u_j} \in E(G) \end{cases}. \tag{2}$$

Then we normalize $str_1(u_i|u_j, G)$ with strengths of relations pointing to u_j and get

$$\overline{str}_1(u_i|u_j, G) = \frac{str_1(u_i|u_j, G)}{\sum_{v \in V(G) and \overrightarrow{vu_j} \in E(G)} str_1(v|u_j, G)} \tag{3}$$

The second definition assumes that initial strength of relation $\overrightarrow{u_i u_j}$ is decided by social circles' similarity and only applies to graphs formed by 2-length links, which is

$$str_2(u_i|u_j, G_2) = card(\{v|v \in V(G_1) \wedge \overrightarrow{vu_i} \in E(G_1) \wedge \overrightarrow{vu_j} \in E(G_1)\}, \tag{4}$$

According to the definition we have $str_2(u_i|u_j, G_2) = str_2(u_j|u_i, G_2)$, $\langle G_1, G_2 \rangle \in \{\langle F_1, F_2 \rangle, \langle V_1, V_2 \rangle\}$. Then we normalize $str_2(u_i|u_j, G)$ with strengths of relations pointing to u_j and get

$$\overline{str}_2(u_i|u_j, G_2) = \frac{str_2(u_i|u_j, G_2)}{\sum_{v \in V(G) and \overrightarrow{vu_j} \in E(G)} str_2(v|u_j, G_2)}. \tag{5}$$

Here we are using the two strengths to see whether online individual relation-network similarity would help to infer interest similarity.

3.5   Interest Prediction

As we discussed in the last section, we use both other users' interests, who are linked to a chosen blogger u in a relation graph G, and the relations' strength to give out a bag of tags he is supposed to use recently, which is defined as $\widehat{tag}(u|G)$. Similar to

the concept of "preference", we use "predict-preference" to describe the possibility that u will like the tag w when the prediction is processed in G, which is defined as

$$\widehat{pref}(w|u,G) = \sum_{v \in V(G)} pref(w|v)\, str(v|u,G). \tag{6}$$

Where $str(v|u,G)$ refers to either $\overline{str}_1(v|u,G)$ or $\overline{str}_2(v|u,G)$ in the calculation.

3.6   Judgment

In this section, we introduce "Error Type Ⅰ" and "Error Type Ⅱ" to clarify the judging criteria. And the error rates of these errors are defined as following.

Error Type Ⅰ 's error rate is $\alpha = \dfrac{\sum_{w \in tag(u) - \overline{tag}(u|G)} pref(w|u)}{\sum_{w \in tag(u)} pref(w|B)}.$ (7)

Error Type Ⅱ 's error rate is $\beta = \dfrac{\sum_{w \in \overline{tag}(u|G) - tag(u)} \widehat{pref}(w|u,G)}{\sum_{w \in \overline{tag}(u|G)} \widehat{pref}(w|u,G)}.$ (8)

In our work, we use "hit-scores" to decide the prediction results, which is defined as

$$mark(u|G) = 1 - p\beta, \tag{9}$$

Where p is a constant and $p > 0$; when $p = 1$, $mark(u|G)$ is the hit ratios of predict results and larger p performs as punishment to Error Type Ⅱ. In the experiments, we simply use $p = 1$. In addition, α, Error Type Ⅰ 's error rate, is not included main criteria, since there are unavoidable individuality of people's interests, and some bloggers employ sparsely used words as tags of their articles. Thus, it is both rough and meaningless attempting to predict all tags.

4   Experiments

In the description of the results, we primarily employ arithmetic mean to measure the overall level of hit scores, and skewness to describe the normality of hit score's distribution. All the skewnesses of hit-score distributions in our experiments are within (-1, 1) and have little negative bias. This means that predict results are overestimated for a bit.

4.1   Dataset

Our experiments at first randomly crawled more than 10k users' data from *blog.sina.com.cn*, including their blogs' tags and their 1-degree friends, 2-degree friends and the co-visited are crawled. As we want to keep only relatively active users in the experiments to avoid random noises in prediction, user with more than 10 blogs, 10 friends, and 40 co-visitors are included and only 584 bloggers meet the qualification. In detail, the concrete numbers result from a trade-off between larger dataset and more information per user. In addition, the process of crawling takes less than one week in order to avoid interest drift.

4.2   Comparison the Predictions

Here we will primarily compare the predictions of different networks as well as their combinations.

Combination of Results. In the experiment, the linear combinations of predict-preferences from F_1 and V_2, and F_2 and V_2 are introduced. And we attach a minus effect to the tag w, if not both of u's friends and co-visited users prefers w. Then we have combine-predict-preference defined in Equation (10).

$$\widehat{pref}(w|u,G,V_2) =$$

$$f\left(\widehat{pref}(w|u,G),\widehat{pref}(w|u,V_2)\right) \cdot \left(a \cdot \widehat{pref}(w|u,G) + (1-a) \cdot \widehat{pref}(w|u,V_2)\right). \quad (10)$$

Where u is a user in the relationship graph G ($G \in \{F_1, F_2\}$) and V_2; w is a tag; a is a constant ranging in (0, 1). And we have $f(x) = \begin{cases} 1, & x \neq 0 \\ c, & x = 0 \end{cases} (c \in [0,1])$.

Preliminary Comparison. We compare the average hit-scores in different relation networks here (showed in the 2nd col. of Table 1). Then, we attempt to exclude tag w from user u's predict-tag set $\widehat{tag}(u|G), G \in \{F_1, F_2, V_2\}$, if $\widehat{pref}(w|u,G) < 0.10$ (The result is showed in the 3rd col. of Table 1). $\overline{str_1}$ is used in this section.

Table 1. means of hit scores in relation networks

Relation network	Average hit-scores	Average hit-scores (with predict-preference threshold)
F_1	0.4573	0.5553
F_2	0.4531	0.5642
V_2	0.4431	0.6419
$F_1 \& V_2$	0.4055	0.5665
$F_2 \& V_2$	0.4365	0.6317

F_1 works better than the others, unless we import predict-preference threshold. Considering the disparity of the two results, we need to find out factors other than various relation networks affect the results, which is discussed in the next section.

4.3   Adjusting the Parameters

Minimal Prediction Preference. We introduce the predict-preference threshold m ranging from 0.00 to 0.90, thus the tags whose predict-preferences are below which are excluded from prediction results. When we compare results in different networks. In all situations, the average hit-scores increase with m. In addition, V_2 and $F_2 \& V_2$ work better than the others. $\overline{str_1}$ is used in this section.

A possible explanation is that when a tag's prediction preference is higher, it is more frequently used in the social network, and it empirically more tends to be a hypernym. However, when we have larger m, valuable information contained in the results is less. For example, "varied opinions" is less meaningful in prediction,

comparing to "food". In addition, α, the error rate of Error Type I , also increases with m, as there are less tags in the results.

Different Strength Definitions. Here we discuss whether different strengths will lead to different level of prediction results. 2-degree friendship network with $\overline{str2}$ is F_2^+, and co-visited relationship network with $\overline{str2}$ is V_2^+.

When we use $\overline{str2}$, average hit-scores still rise as m grows. And $\overline{str1}$ works better than $\overline{str2}$ in most situations. Therefore the number of common vertexes in social networks might not be in charge of social influence among users.

Parameter a and c. Here we use m=0.5 to balance the level of hit scores and information content in prediction. We combine prediction results in F_1 and V_2, F_2 and V_2 separately. $\overline{str1}$ is used in this section.
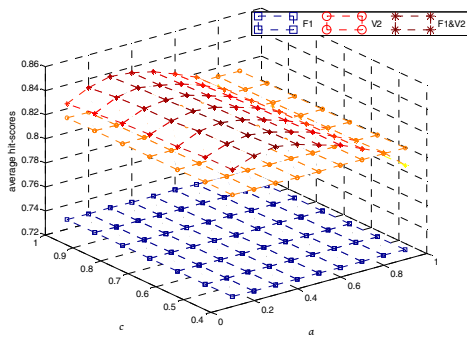
Fig. 1. Linear combination of F_1 and V_2's results with different a and c
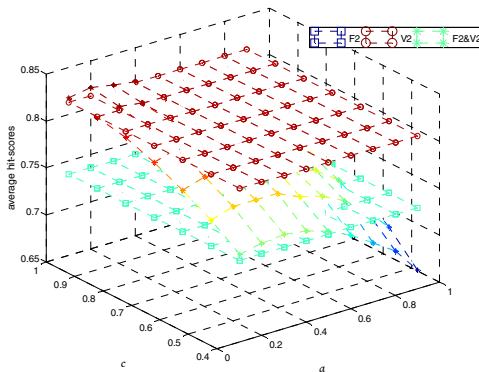
Fig. 2. Linear combination of F_2 and V_2's results with different a and c

In the second combination the minus-factor c does not help (showed in Fig 2), unlike the first one (showed in Fig 2). A possible explanation is that friends and co-visitors' contents are directly connected with the users', while 2-degree friends have friends in mediation, where bias appears.

5   Conclusion

In this paper, we present a study focusing on different effectiveness of two widely used assumptions in user interest prediction in social networks. One assumption is that people show similar interests with their graphic neighbors. And the other is that alike users share alike preferences, based on which we have a deduction that contents favored by the same user tend to be similar. The predications are carried out in relation networks consist of friends (F_1), 2-degree friends (F_2), the co-visited users (V_2), and their combinations ($F_1\&V_2$, $F_2\&V_2$) separately. With changing parameters, we find that V_2 and $F_1\&V_2$ generally work better.

As for the parameters, we find that we will get higher hit-scores but less information with a higher filter threshold, in all cases. Also, relation strength based on the number of common friends or common visitors does no help in this work. After that, when we compare the prediction results of $F_1\&V_2$ and $F_2\&V_2$, the two results show quite different changing properties with linear coefficient and minus-effect coefficient.

In the future, we plan to expand our models with different user behaviors. Also, we will attempt to conduct prediction on a network containing more complex relationships and do test on different blog communities.

References

1. Agichtein, E., Brill, E., Dumais, S.T.: Improving Web Search Ranking by Incorporating User Behavior Information. In: SIGIR 2006, pp. 19–26 (2006)
2. Tsai, P.-Y., Liu, D.-R.: Personalized Popular Blog Recommender Service for Mobile Applications. In: Di Noia, T., Buccafurri, F. (eds.) EC-Web 2009. LNCS, vol. 5692, pp. 2–13. Springer, Heidelberg (2009)
3. Schenkel, R., Crecelius, T., Kacimi, M., Neumann, T., Parreira, J.X., Spaniol, M., Weikum, G.: Social Wisdom for Search and Recommendation. IEEE Data Eng. Bull., 40–49 (2008)
4. Wen, Z., Lin, C.-Y.: On the Quality of Inferring Interests from Social Neighbors. In: KDD 2010 (2010)
5. Shepitsen, A., Gemmell, J., Mobasher, B., Burke, R.: Personalized Recommendation in Social Tagging Systems Using Hierarchical Clustering. In: RecSys 2008 (2008)
6. Stoyanovich, J., Yahia, S.A., Marlow, C., Yu, C.: Leveraging Tagging to Model User Interests in del.icio.us. In: Proceedings of the 2008 AAAI Social Information Spring Symposium (2008)
7. Crandall, D., Cosley, D., Huttenlocher, D., Kleinberg, J., Suri, S.: Feedback Effects between Similarity and Social Influence in Online Communities. In: KDD 2008 (2008)
8. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM 35(12), 61–70 (1992)
9. Li, X., Guo, L., Zhao, Y.: Tag-Based Social Interest Discovery. In: WWW 2008 (2008)
10. Song, X., Tseng, B.L., Lin, C.-Y., Sun, M.-T.: Expertisenet: Relational and Evolutionary Expert Modeling. In: UM 2008 (2008)
11. Borgatti, S.P., Mehra, A., Brass, D.J., Labianca, G.: Network Analysis in the Social Sciences. Science 323, 892–895 (2009)
12. White, R.W., Bailey, P., Chen, L.: Predicting User Interests from Contextual Information. In: SIGIR 2009, pp. 363–370 (2009)
13. Lu, D., Li, Q.: Personalized Search on Flickr Based on Searcher's Preference Prediction. In: WWW 2011 (2011)

P2P-Based Publication and Sharing of Axioms in OWL Ontologies for SPARQL Query Processing in Distributed Environment

Huayou Si[1,2], Zhong Chen[1,2], Yun Zhao[3], and Yong Deng[1,2]

[1] Software Institute, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China
[2] Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing 100871, China
[3] School of Information Engineering, Zhejiang Agriculture and Forestry University, Hangzhou 311300, China
{sihy,chen}@infosec.pku.edu.cn

Abstract. OWL ontologies are used to represent knowledge in web. In recent years, with the wide application of Semantic Web, large numbers of OWL ontologies have appeared on Internet, especially, in some virtual knowledge communities. These ontologies are distributed in different sites and provide an amount of knowledge to query. But, it has become a pressing issue that, given a semantic query, how to efficiently gather the related knowledge from these ontologies located in different sites to process it. To address this issue, in this paper, we propose a P2P-based approach to publish axioms in sharable ontologies and freely sharing them in an open distributed environment. Given a query of SPARQL, this approach can automatically gather published axioms related to the query to process the query. As a knowledge sharing approach, our approach can directly share axioms coming from different ontologies on different nodes. It overcomes limitations of those approaches which focus on how to locate related ontologies for a query. We also conducted two experiments to evaluate the effectiveness and the efficiency of our approach. The experimental results demonstrated that it is effective and efficient.

Keywords: OWL Ontology, Ontology Axiom, SPARQL Query, P2P.

1   Introduction

In recent years, Web Ontology Language (OWL) [1] has been brought into wide use to create OWL ontologies for representing knowledge in web. Large numbers of sharable OWL ontologies have appeared on the Internet. These ontologies possess a large quantity of knowledge. They can serve as sources of knowledge for web-based question answering system, recommendation system, etc [2, 3]. In addition, in some virtual knowledge communities [4, 5], numbers of OWL ontologies are also created. They also possess a large quantity of knowledge to be shared and leveraged by members in the community for their own purposes.

But, how to effectively share knowledge in ontologies in an open distributed environment, especially within a virtual community, has become a pressing issue. It has been given a great deal of attention in practice as well as in research. Most of current approaches are based on Client–Server (C/S) structure. In these approaches, all knowledge resources, i.e., ontologies, are gathered and stored in some centralized knowledge servers. Users can query and utilize knowledge under some sort of centralized control. These approaches have been considered inappropriate and ineffective to share knowledge [3, 6] and are not suitable for the autonomous and dynamic characteristics of knowledge sharing [7, 8]. So, knowledge sharing in a decentralized network, especially supported by peer-to-peer (P2P) technology, is introduced. Given a query, these approaches either route it to the nodes with related ontologies to construct solutions for it, or locate the related ontologies and then send the query to them to construct solutions respectively. In these approaches, knowledge sharing is based on ontology, not on the knowledge itself in ontology. So, these approaches do not run well in some cases, such as, though we can not construct any solutions for a query from ontology *A* or ontology *B* respectively, yet we can from knowledge together in ontology *A* and *B*; or we can construct more solutions from knowledge together in ontology *A* and *B* than from ontology *A* and *B* respectively.

To address the issue and overcome the limitations of current approaches, we propose a distributed approach to directly publish and share axioms in OWL ontology and implement it based on structured P2P network. In our approach, if a node has sharable OWL ontologies, it can publish all the axioms in these ontologies. If a node receives a query of SPARQL (Simple Protocol and RDF Query Language) [9], it can efficiently retrieves the axioms related to the query and creates a temporary ontology to process it. Knowledge sharing in our approach is based on each axiom in OWL ontology rather than entire ontology. If necessary, related axioms in different ontologies locating in different nodes can be gathered to process a query.

We also conducted two experiments to evaluate the effectiveness and the efficiency of our approach. The experimental results demonstrated that our approach is effective and efficient.

2   Basic Idea and Overview of Our Approach

In this section, we first discuss our basic idea and then outline our approach.

2.1   Basic Idea

By W3C Recommendation [1], OWL ontology is a formal description of a given domain, which consists of the following three different syntactic categories:

- *Entities*, such as classes, properties, and individuals, are identified by IRIs. They form the primitive terms of ontology to express the basic notions in domain. For example, the class *http://www.example.com/onto.owl#Person* represents the set of all people. It can be abbreviated as *a:Person*, where *a:* denotes the name space: *http://www.example.com/onto.owl#*.
- *Expressions* represent complex notions by using entities. For example, person with three children is a class expression defined based on *person* and *child*.
- *Axioms* are statements that are asserted to be true in domain being described.

For example, using a subclass axiom, one can state that class *a:Student* is a subclass of class *a:Person*. In Functional-Style Syntax [1], the axiom is represented as: *SubClassOf(a:Student a:Person)*. In fact, axioms use entities and expressions to state the explicit facts in domain. An axiom, where one or more entities appear in it, is a piece of knowledge in domain being described.

These three syntactic categories are used to express the logical part of OWL ontologies, from which useful inferences can be drawn. Because the assertions of entities and expressions are also axioms, OWL ontology is essentially a set of axioms. So, given an OWL ontology, we can take out all its axioms. For each axiom, we view each entity appearing in it as an index of the axiom. Then, based on each index of an axiom, we can publish and retrieve it on P2P network. This is our idea to publish axioms and share them. We will discuss its implementation in detail in subsection 2.2.

In addition, as another recommendation of W3C, SPARQL [9] is a query language for RDF graphs. Each SPARQL query has a graph pattern, from which, we can extract a set of RDF terms. Graph pattern is a core component of SPARQL query. It is used to match a sub-graph of the RDF data being queried when RDF terms from that sub-graph may be substituted for the variables in graph pattern, and the result is RDF graph equivalent to the sub-graph. So, given a SPARQL query, if a RDF graph can be queried out results for the query, RDF terms appearing in its graph pattern must be appear in the RDF graph. Accordingly, given a query, OWL ontology axioms related to the RDF terms, which appear in the query's graph pattern, are usually useful to process the query. Here, it should be noted that OWL ontology is based on RDF data model. OWL ontology is essentially RDF graph. Moreover, each axiom can be converted into a semantically equivalent set of triples. Entities in OWL ontology are called RDF terms in RDF data.

Thus, based on the foregoing idea of axiom publication and retrieving, given a SPARQL query, we can retrieve related axioms for the query based on the entities, i.e., RDF terms, which appear in the query's graph pattern.

2.2   Overview of Our Approach

Peer-to-peer (P2P) systems usually consist of large numbers of autonomous nodes and allow the sharable resources of each node to be accessed by others. Especially structured P2P systems, such as Chord [10], they usually organize nodes in a systematic way and publish every sharable resource to a given node respectively. As a result, they can effectively find out a given resource and provide very good scalability. Because of structured P2P with these strengths, we apply it to our approach for knowledge sharing. Based on structured P2P protocols, we can design two functions to publish and retrieve axioms in OWL ontology on P2P network as follows:

- ***pubAxiom(Entity, Axiom)***, the function is used to publish *Axiom* based on *Entity*. According to a given structured P2P protocol, it locates a given node *N* based on *Entity* so as to saves the pair <*Entity*, *Axiom*> on node *N*.
- ***retrAxiom(Entity)***, the function is used to get all the axioms that are published based on *Entity* by any node. According to a given structured P2P protocol, it locates a given node *N* based on *Entity* and retrieve the pairs <*Entity*, *Axiom*> saved on node *N*. And then get axioms concerned.

In our approach, the nodes concerned constitute a structured P2P network. When a node with some sharable ontologies joins, it publishes them as follows:

1. For each sharable ontology *O*, extracts all axioms in it as a set *axSet*.
2. For each axiom *ax* in set *axSet*, parses out all entities as a set *enSet* from *ax*.
3. For each entity *en* in set *enSet*, publishes axiom *ax* based on entity *en* by using function *pubAxiom*(*en*, *ax*) as mentioned above.

Once a node receives a SPARQL query, according to the idea as discussed in subsection 2.1, it processes the query as follows:

1. Parses the query's graph pattern and extracts all the entities as a set *enSet*.
2. Based on some strategies and using function *retrAxiom*(*Entity*), retrieves enough related axioms for the query according to entities in set *enSet*.
3. Creates a temporary ontology *O* which holds all the axioms being retrieved.
4. Reasons ontology *O* to construct solutions for the query.

The strategy to retrieve related axioms is discussed in the following section.

3 Definitions and Axiom Retrieving Process

In this section, first we present several related definitions and conclusions and then outline the process of axiom retrieving of our approach.

3.1 Definitions and Conclusions

To conveniently present the process of axiom retrieving for processing a SPARQL query, here we give several definitions as follows:

- **Direct Relevance Axiom**: If an entity *en* appears in an axiom *ax*, axiom *ax* is called Direct Relevance Axiom of entity *en*.
- **Level 1 Relevance Axioms (LRA 1)**: Given an entity set *enSet*, the axioms in set *axSet*, which consists of all the Direct Relevance Axioms of any entity in set *enSet*, is called Level 1 Relevance Axioms (LRA 1) of entity set *enSet*.
- **Level k Relevance Axioms (LRA k)**: Given an entity set *enSet*, all the Direct Relevance Axioms of any entity in entity set *entitySet* is called Level k Relevance Axioms (LRA k) of set *enSet*, where entity set *entitySet* consists of all the entities which appear in any axiom in LRA k-1 of entity set *enSet*.
- **Relevance Axiom Closure (RAC)**: In a given context and for an entity set *enSet*, if LRA k is identical to LRA k+1 and not to LRA k-1. The LRA k is called Relevance Axiom Closure (RAC) of entity set *enSet*.

If all the entities in set *enSet* come from graph pattern of a SPARQL query *Q*, the Relevance Axioms of set *axSet* is also called Relevance Axioms of query *Q*. Given a context, if all the axioms coming from different ontologies are consistent, then based on the semantics of OWL we can draw two conclusions as follows:

- **Conclusion 1:** Given a SPARQL query *Q*, solutions, which are constructed for query *Q* based on LRA k+1 of query *Q*, are not less than the solutions based on LRA k of query *Q*.

OWL ontology is based on Description Logic, a subset of first-order logic, which is Monotonic Logic. Thus, if a solution can be reason out from LRA k, surely it can reason out from LRA k+1, because LRA k+1 includes LRA k.

- **Conclusion 2:** Given a SPARQL query *Q*, if *Q*'s RAC can be gotten in a given context, all potential solutions can be reasoned out for the query *Q*.

Relevance Axiom Closure (RAC) includes all direct and indirect axioms related to query *Q*, just from which solutions of query *Q* can be reasoned out. The axioms, which are not included in RAC, cannot contribute to construct solutions for query *Q*.

3.2   Axiom Retrieving Process of Our Approach

Based on function *retrAxiom*(*Entity*) discussed in section 2.2, we design a algorithm *retrieveRldAxioms* shown in Figure 1 to get Relevance Axiom Closure (RAC) for an entity set *enSet* of a SPARQL query. The basic idea is that: given an entity set, first we retrieve its LRA 1 (Level 1 Relevance Axioms), and then based on the new entities introduced by LRA 1 we retrieve its LRA 2. Repeat the strategy until we can not get more axioms. Thus, all the retrieved axioms constitute the query's Relevance Axiom Closure.

```
1.       Algorithm retrieveRldAxioms
2.       Input enSet: the Set of entities parsed out from graph pattern of a SPARQL query
3.       Output axSet: the axiom Set retrieved by the algorithm
4.       initiates Lists levelA, levelB, current, and constru to empty;
5.       puts all the elements in enSet into levelA;
6.       current= levelA; constru= levelB;
7.       while(true){
8.         for(each entity en in List current){
9.           retrieves a Set axiomSet using function retrAxiom(en) defined in section 2.2;
10.          puts all the axioms in axiomSet into axSet;
11.          for(each axiom ax in axiom Set axiomSet){
12.            parses axiom ax to get all entities appearing in ax as a Set entitySet;
13.            for(each entity ent in Set entitySet){
14.              if(ent is not an element of enSet) puts ent into Set enSet and List constru; }
15.          }
16.        }
17.        if(List constru is empty) break;
18.        if(current = =levelA){ current=levelB; constru=levelA;}
19.        else{current=levelA; constru=levelB;}
20.        sets List constru empty;
21.      } return axSet;
```

Fig. 1. Algorithm: *retrieveRldAxioms*

Though we can reason out all potential solutions for a query based its Relevance Axiom Closure according to Conclusion 2, it may be difficult to get the Relevance Axiom Closure in large-scale knowledge communities because there may be a mass of related axioms and need great numbers of accesses to P2P network, i.e., calls of function *retrAxiom(Entity)*. In practice, users sometimes do not need all possible solutions of their queries. Thus, for a query we can just retrieve Relevance Axioms of a given Level.

4   Evaluation

We design two experiments to evaluate our approach's effectiveness and efficiency. The first experiment evaluates its ability of obtaining potential solutions for a given query. The second experiment reveals the specific effects of a query's different LRAs (Level Relevance Axioms) on obtaining potential solutions for the query to gain a better understanding of the efficiency of our approach.

To evaluate our approach, we have implemented it. In our experiments, we use the OWL ontologies and SPARQL queries of the third party as experimental data, which locate in "pellet-2.2.2\examples\data" in the open source development kits "pellet-2.2.2". It provides several OWL ontologies and dozens of SPARQL queries. First of all, we should find out the number of all the potential solutions of each query in our experimental data. First, we merge all OWL ontologies in our experimental data as a new ontology O. Then, for each query, we reason ontology O to find out the number of its solutions. Obviously, the number is the query's potential solution number.

Experiment 1: In this experiment, to simulate a distributed virtual community, first we merge all OWL ontologies in our experimental data as a new ontology O. Then, randomly divide the axioms in ontology O into several ontologies and deploy them on different nodes. Here, we conduct 10 tests. For the first one, we directly deploy ontology O. For the second one, we divide it into 2 ontologies. Until for the 10th one, we divide it into 10 ontologies. For each test, we use our approach (M1) to process all our queries and calculate the percentage of the returned solutions together of all queries. Then, based on idea in [5, 7] which locate related ontologies on different nodes and send query to them to construct solutions respectively, we implement approach (M2) to do the same thing. The results are shown in Figure 2.
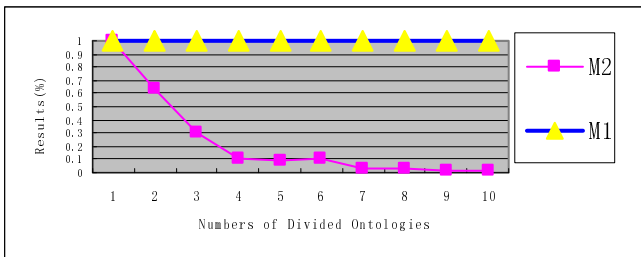
Fig. 2. Comparison of Callback Ability of Our Approach M1 to Typical Approach M2

Figure 2 shows that, for approach M2, we divide ontology O into more ontologies, less solutions can be reasoned out. However, our approach M1 can always get all the potential solutions for all queries whatever ontology O is divided. It is because, for a query, although related axioms are divided into different ontologies, M1 can retrieve them and put them together to process the query. However, M2 can not. M2 shares knowledge based on located ontology itself. Thus, axioms are divided into more ontologies, related axioms are more scattered, less solutions can be gotten by M2.

Experiment 2: In this experiment, first we publish all OWL ontologies in our experimental data using our approach. Then, for each query, we retrieve its LRAs of all possible Levels respectively. Next, for each query q, we get the numbers of query

q's solutions respectively based on each LRA of query q. Finally, for each Level k, we record the percentage of queries, which potential solutions can be reasoned out completely based on their own LRA k. The experimental results are shown in Figure 3. Based on LRA 1, 25% of queries get all its potential solutions. Based on LRA 2, the percentage is 100%. They mean that, given a query, it is highly possible to get all its potential solutions just based on its LRA 2. It because that the semantics of an entity et is mainly associated with the entities (as set $enSet$), which sometimes appear in some Direct Relevance Axiom of entity et. LRA 2 contains all the axioms which are Direct Relevance Axioms of an entity in set $enSet$. So, the results are reasonable.
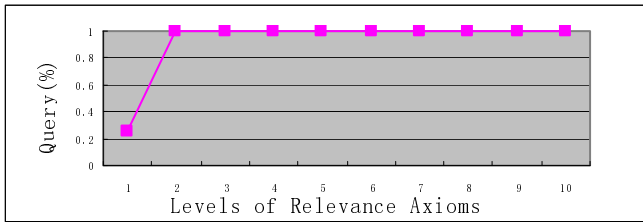
Fig. 3. Percentages of Queries Getting All Potential Solutions According to Each LRA

From the two experiments, it can be seen that our approach is effective and efficient. If necessary, it can get all potential solutions for a query. Usually it is unnecessary to retrieve its Relevance Axiom Closure even if all solutions are needed.

5   Related Work

Along with the development of P2P technique and the technical requirement of knowledge sharing, in recent years, some P2P-based approaches for ontology publication and discovery have been proposed.

Raul Palma1 and Peter Haase [11] provide an approach (named Oyster) to exchange and re-use ontologies based on P2P network. It does not involve in the specific knowledge in ontologies when publishing and discovering them. Chen et al. [7], propose a knowledge sharing approach, which organizes the nodes with sharable knowledge as an unstructured P2P network. For a query, the approach tries to send it to the nodes with related knowledge. Similar approaches are also presented in [3, 5, 8]. In these approaches, knowledge sharing is based on sharable ontology rather than knowledge itself. Min et al. [12] present a scalable distributed RDF repository (named RDFPeers) that stores each triple at three places by applying globally known hash functions to its subject, predicate, and object. Thus, all nodes know which node is responsible for triples they are looking for. Queries are guaranteed to find out matched triples in the network if the triples exist. However, if RDF triples are directly published on P2P, it does not support semantic retrieval. To address this question, Kohigashi et al. [13] focus on class hierarchies of RDF resources, encode them into related resource ID, and publish the ID. And then, they present a P2P information sharing method for RDF triples based on the class hierarchies. However, except for classification information, the method can not yet support complex semantic query.

6   Conclusion and Future Work

In this paper, we propose a P2P-based approach to publish and share axioms in OWL ontologies so as to gather related axioms from different ontologies in different nodes to process SPARQL queries in an open distributed environment. It overcomes the limitation that knowledge sharing is based on entire ontology. We also conducted experiments which results demonstrated that it is effective and efficient. In near future, we plan to continue our research work in the following aspects:

- Conduct experiments for our approach in a large-scale community.
- Deal with the inconsistency among retrieved axioms. In a large-scale community, axioms related to a query maybe come from different ontologies in different nodes. Inconsistency is unavoidable.
- Study the method to map a word to existing ontological entities so as to facilitate requestors to construct their SPARQL queries automatically.

References

1. OWL, http://www.w3.org/TR/owl2-overview/ (retrieved March 17, 2011)
2. Zhong, N., Liu, J., Yao, Y.: Advances in Web Intelligence, p. 239. Higher Education Press, Beijing (2011)
3. Zhen, L., Jiang, Z., Song, H.: Distributed recommender for peer-to-peer knowledge sharing. Information Sciences 180, 3546–3561 (2010)
4. Maret, P., Hammond, M., Calmet, J.: Virtual Knowledge Communities for Corporate Knowledge Issues. In: Gleizes, M.-P., Omicini, A., Zambonelli, F. (eds.) ESAW 2004. LNCS (LNAI), vol. 3451, pp. 33–44. Springer, Heidelberg (2005)
5. Gnasa, M., Alda, S., Grigull, J., Cremers, A.B.: Towards Virtual Knowledge Communities in Peer-to-Peer Networks. In: Callan, J., Crestani, F., Sanderson, M. (eds.) SIGIR 2003 Ws Distributed IR 2003. LNCS, vol. 2924, pp. 143–155. Springer, Heidelberg (2004)
6. Kwok, J.S.H., Gao, S.: Knowledge sharing community in P2P network: a study of motivational perspective. Journal of Knowledge Management 8(1), 94–102 (2004)
7. Wang, C.-Y., Yang, H.-Y., Chou, S.-C.T.: Using peer-to-peer technology for knowledge sharing in communities of practices. Decision Support Systems 45, 528–540 (2008)
8. Jain, L.C., Nguyen, N.T.: Knowledge Processing and Decision Making in Agent-based Systems. Springer (2009)
9. SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/ (retrieved March 18, 2011)
10. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on Networking (TON) 11(1), 17–32 (2003)
11. Palma, R., Haase, P.: Oyster - Sharing and Re-using Ontologies in a Peer-to-Peer Community. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 1059–1062. Springer, Heidelberg (2005)
12. Cai, M., Frank, M.: RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network. In: Proceedings of the 13th International Conference on Word Wide Web (WWW 2004), pp. 650–657. ACM Press (2004)
13. Kohigashi, K., Takahashi, K., Harumoto, K., Nishio, S.: A Peer-to-Peer Information Sharing Method for RDF Triples Based on RDF Schema. In: Omatu, S., Rocha, M.P., Bravo, J., Fernández, F., Corchado, E., Bustillo, A., Corchado, J.M. (eds.) IWANN 2009. LNCS, vol. 5518, pp. 646–650. Springer, Heidelberg (2009)

When Sparsity Meets Noise in Collaborative Filtering

Biyun Hu[1,2], Zhoujun Li[1,2], and Wenhan Chao[1,2]

[1] State Key Laboratory of Software Development Environment, Beihang University, China
[2] School of Computer Science and Engineering, Beihang University, Beijing, China
byhu8210@gmail.com, lizj@buaa.edu.cn, cwh2k@163.com

Abstract. Traditionally, it is often assumed that data sparsity is a big problem of user-based collaborative filtering algorithm. However, the analysis is based only on data quantity without considering data quality, which is an important characteristic of data, sparse high quality data may be good for the algorithm, thus, the analysis is one-sided. In this paper, the effects of training ratings with different levels of sparsity on recommendation quality are first investigated on a real world dataset. Preliminary experimental results show that data sparsity can have po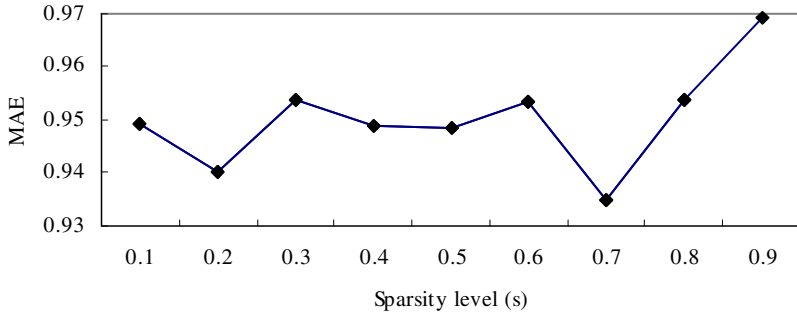sitive effects on both recommendation accuracy and coverage. Next, the measurement of data noise is introduced. Then, taking data noise into consideration, the effects of data sparsity on the recommendation quality of the algorithm are re-evaluated. Experimental results show that if sparsity implies high data quality (low noise), then sparsity is good for both recommendation accuracy and coverage. This result has shown that the traditional analysis about the effect of data sparsity is one-sided, and has the implication that recommendation quality can be improved substantially by choosing high quality data.

Keywords: Collaborative Filtering, Sparsity, Noise, Accuracy, Coverage.

1   Introduction

Traditionally, data sparsity is seen as a big problem of user-based CF [1][2][3][4][5]. It is often claimed that, data sparsity results in not enough co-rated items between two users, incurring unreliable user similarity information, and further, poor recommendation accuracy. Even more, there may be no co-rated items between the active user and other ones, thus no recommendations can be made for the active user, causing poor recommendation coverage. The main disadvantage of these analyses is that only data quantity is considered. Another important characteristic of ratings, i.e., data quality is not taken into consideration. Sparse data, but with high quality (low noise), may be good for recommendation quality. Thus, these analyses are one-sided. Bobadilla et al.'s [4] experimental results also show that these analyses are one-sided. They have evaluated the recommendation quality of user-based CF algorithm on training ratings with different levels of sparsity. Experimental results show that, as the level of data sparsity increases, the recommendation accuracy of user-based CF (using Pearson correlation coefficient as the similarity method) even increases. However, they didn't give an explanation to this interesting result.

To comprehensively investigate the effects of data sparsity on user-based CF algorithm, this paper first evaluates the recommendation quality of user-based CF with ratings having different levels of data sparsity. After obtaining preliminary results, we describe the computation of noise in ratings, and further choose data with different levels of sparsity and noise to comprehensively evaluate the effects of sparsity.

2   Collaborative Filtering and Data Sparsity

2.1   Collaborative Filtering

A typical CF system has a set of users U = { $u_1, u_2, ..., u_M$ }, a set of items I = { $i_1, i_2, ..., i_N$ }, and a series of ratings $R = \{r_{u,i} \mid u \in U, i \in I\}$ denoting user u's rating for item i. A rating can be integral or continuous. The task of CF is to predict users' preference for unrated items, so then a list of most preferred items can be recommended to users.

To improve recommendation quality, many CF algorithms have been proposed [2] [3] [6] [7] [8] [9]. One of the most well-known CF algorithms is the user-based CF algorithm proposed by Resnick et al. [2]. The basic idea of it is that people expressed similar preferences in the past will prefer similar items in the future. Used-based CF has three key steps. First, the similarity between any two users is measured. Pearson coefficient and vector similarity can be used for the measurement. Literature has shown that Pearson coefficient is widely used [1]. The coefficient between any two users is computed based on the two's ratings awarded to the co-rated items. Then, a number of users having the highest similarity values with the active user are selected as neighbors. Finally, a prediction for the active user on the target item is given according to the neighbors' ratings. User-based CF models the way of social recommendation, that is, ask friends, and get recommendations, so it is good at communicating reasonably with users [10]. In the followings, user-based CF is denoted as CF for short.

2.2   Data Sparsity Problem

In many commercial recommendation systems, even active users may have rated well under 1% of the items, causing the data sparsity problem [3]. Traditionally, it is often analyzed that CF is susceptible to data sparsity, because the sparsity may refrain it from finding out reliable and enough neighbors [3] [4], resulting in poor recommendation accuracy and coverage. However, the analysis is made based only on rating quantity. Data quality, an important property of ratings, is totally neglected, sparse high quality data may be good for recommendation quality, thus, the analysis is one-sided. In support of this, we cite the experimental results reported by Bobadilla et al. [4]. For the first time, they have made a detailed experimental investigation into the effects of data sparsity on the recommendation quality of CF. Their results show

that, as the level of data sparsity increases, the recommendation accuracy of CF even gets better. Our experimental results reported below also show that data sparsity can have positive effects on the recommendation quality.

3   Experimental Investigation on the Effects of Data Sparsity

3.1   Dataset

The MovieLens dataset, one of the most often used datasets in the CF research field, provided by the MovieLens recommendation system [3] is used in the experiments. The dataset contains 100,000 ratings of approximately 1,682 movies made by 943 users. Ratings are discrete values from 1 to 5. Each user has at least 20 ratings. As Sarwar et al. [3] do, we randomly split 80% of the dataset into a training set and the remaining into a test set.

3.2   Procedures

To investigate the effects of data sparsity, as Bobadilla et al. do [4], we first random select 90%, 80%, …, and 10% of the ratings in the original training set to form new training sets with different levels of sparsity s: 0.1, 0.2, … , and 0.9 (i.e., $s = 1$ - (percentage of ratings chosen from the original training set)). The bigger the s, the sparser the corresponding new training set is. Then, CF algorithm is used on each new training set to predict the ratings in the test set. Pearson correlation coefficient [1] is used to compute the similarity between any two users. The neighborhood size is set 5, 10, … , and 60 respectively. Rating prediction is given as the weighted sum of the neighbors' ratings [1].

3.3   Metrics

Recommendation quality refers to recommendation *accuracy* and *coverage* in this paper. The former is most often investigated [11] It measures how well predictions given by CF algorithms approximate to actual ratings in the test set. Popular used accuracy metrics include MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error) [10]. MAE is used in our experiments.

Recommendation coverage evaluates how many ratings in the test set can be predicted. It is less investigated than recommendation accuracy; however, it is an important quality metric, because systems with lower coverage may be less valuable to users [10]. Recommendation coverage is often measured by the ratio of predicted ratings to all the ratings N in the test set [12].

3.4   Results

Averaged results are reported in this paper (over those obtained when the number of neighbors is set 5, 10, …, and 60 respectively). MAE and Coverage results under different levels of sparsity are reported in Figs. 1 and 2.

Fig. 1 shows that when the sparsity level *s* increases from 0.1 to 0.2, from 0.3 to 0.4, and from 0.6 to 0.7, MAE decreases, i.e., recommendation accuracy increases. Fig. 2 shows that, recommendation coverage keeps increasing as the sparsity level ascending from 0.1 to 0.7. Recommendation accuracy decreases when the sparsity level increases from 0.2 to 0.3, from 0.5 to 0.6, and from 0.7 to 0.9 (Fig. 1). Recommendation coverage drops when the sparsity level increases from 0.8 to 0.9 (Fig. 2). In all, experimental results show that *data sparsity can have positive effects on both recommendation accuracy and coverage*.

Fig. 1. With the different levels of data sparsity *s*, the recommendation accuracy results (*MAE*)
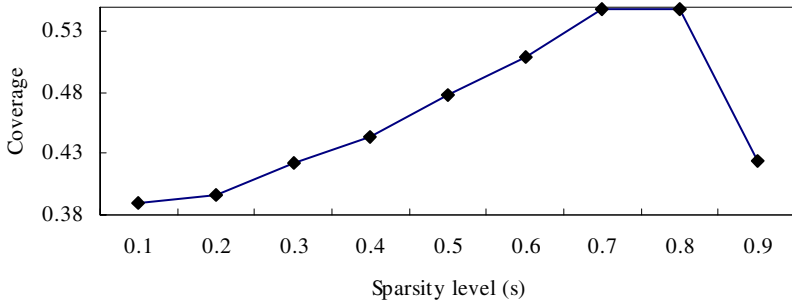
Fig. 2. With the different levels of sparsity *s*, the recommendation *coverage* results

4   Considering Noise, A Re-investigation of the Effects of Sparsity

Bobadilla et al.'s and our experimental results both show that data sparsity can have positive effects on the CF algorithm. This confirms that it is one-sided to think that data sparsity is a big problem of CF. We think that when considering the effects of data sparsity, data noise should also be taken into consideration, because sparse high quality data may be good for recommendation quality. To verify this, in the followings, the computation of the noise in ratings is first described, and then a re-investigation of the effects of data sparsity with data noise taken into consideration is presented.

4.1   Noise

Users' ratings manifest how well those items meet their interests, however, during the rating process, factors unrelated to user interests, such as typos, inconsistent usage of a rating scale or careless rating, may add noise to ratings. In our previous work, it is shown that users' latent preference is more accurate than rating in representing user preference [13], thus, in this paper, noise in a rating is denoted as:

$$n_{u,i} = |r_{u,i} - l_{u,i}| \tag{1}$$

where $n_{u,i}$ is the noise in the rating $r_{u,i}$, and $l_{u,i}$ is the latent preference for user u on item i. The key step for calculating the noise in a rating is to obtain latent preference. The computation of latent preference can be found in the previous work [13].

4.2   Re-investigation of the Effects of Sparsity

Firstly, according to the process described in the previous work [13], for each rating in the origin training set, the corresponding latent preference and noise are computed with Formula (1).

To verify that sparse high quality data is good for recommendation quality, we have designed two groups of experiments where low data quality and high quality are respectively implied by data sparsity.

For the first group, we have chosen the original training ratings with noise $n_{u,i}$ ranging from n_1 to n_2 to form new training sets (n_1 is set 0, 0.5, … , and 2.5 respectively, and n_2 is set 0.5, 1, … , and 3 respectively). The corresponding sparsity level s is 0.58, 0.68, 0.83, 0.94, 0.98, and 0.99. The experiments are designed in such a way for the reason that, noise distribution analysis shows, the higher the noise, the fewer the ratings with the noise. The recommendation quality results on these new training sets are reported in Figs. 3 and 4. The two figures show that, as the sparsity level s increases, the noise level n ascends, MAE increases and Coverage decreases. These experimental results show that, *when data sparsity means low data quality, recommendation accuracy and coverage decrease*.
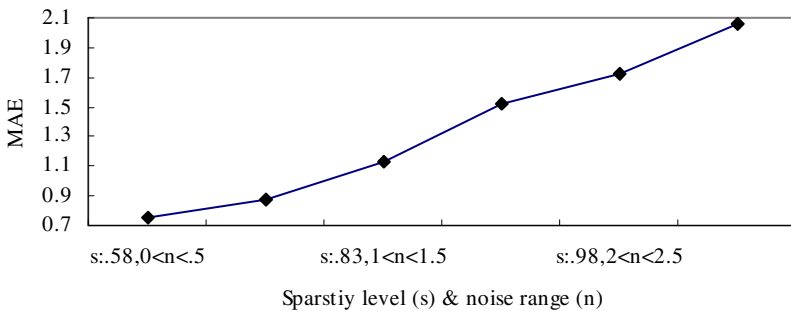
Fig. 3. Sparsity s means low data quality (high noise n). MAE increases as s increases.
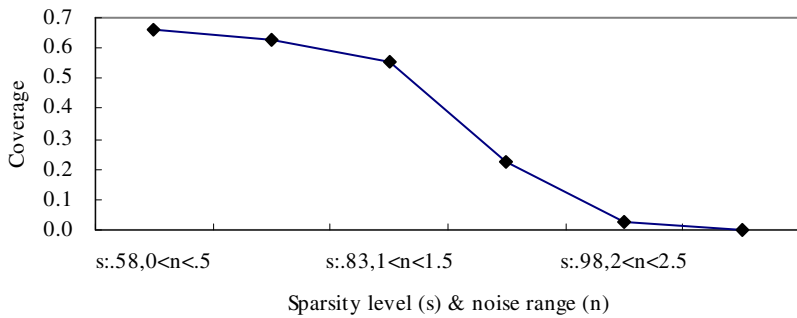
Fig. 4. Sparsity *s* means low data quality (high noise *n*). Coverage decreases as *s* increases.

For the second group of experiments, we have chosen the original training ratings with noise $n_{u,i}$ smaller than n to form new training sets (n is set 3, 2.5, … , and 0.5 respectively). The corresponding sparsity level s is 0, 0.01, 0.03, 0.19, 0.26, and 0.58 respectively. These new training sets are then used by the CF algorithm to make predictions for the test ratings. The recommendation quality results on these new training sets are reported in Figs. 5 and 6, which show that, as the sparsity level s increases, the noise level n decreases, MAE drops, and Coverage increases. These results mean that, *when data sparsity implies high data quality, recommendation accuracy and coverage increase.*

Compared with the recommendation accuracy (MAE: 0.9) and Coverage (0.44) obtained on the original training set, those training ratings with sparsity level *s*: 0.58 and noise $0 < n < 0.5$ produce the recommendation results with MAE 0.76 and Coverage 0.66 (shown in Figs. 5 and 6). This means that *sparse high quality data can improve recommendation accuracy and coverage by 15.6% and 50% respectively.*
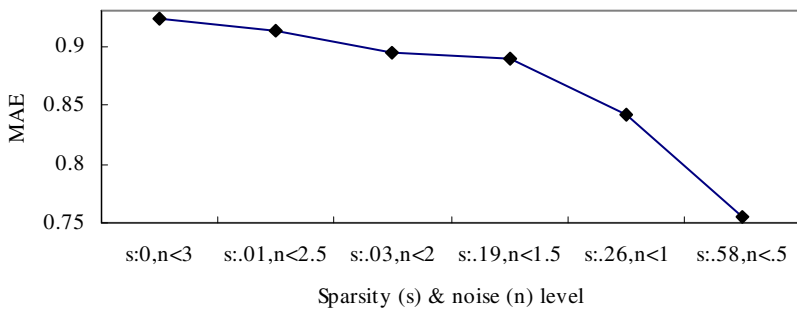
Fig. 5. Sparsity *s* implies high data quality (low noise *n*). MAE decreases as *s* increases.
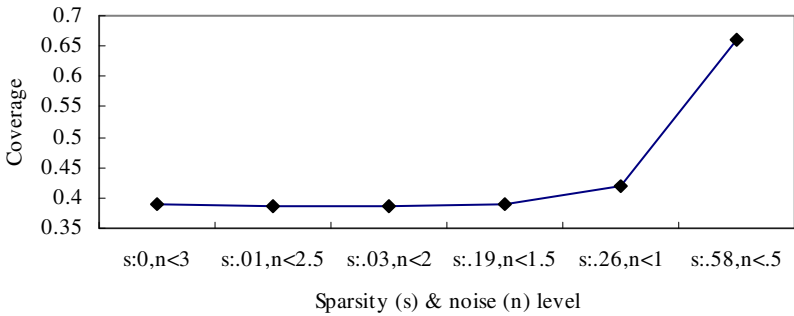
Fig. 6. Sparsity *s* implies high data quality (low noise *n*). Coverage increases as *s* increases.

5   Conclusion

The main contribution of this work is that, regarding the effects of data sparsity on the recommendation quality of user-based collaborative filtering, experimental results different from traditional analysis are shown. These are: a) data sparsity can have positive effects on both recommendation accuracy and recommendation coverage; b) sparse high quality data can improve recommendation quality significantly. These results have shown that: a) the traditional analysis about the effects of data sparsity on the user-based collaborative filtering is one-sided; and b) recommendation quality can be improved by choosing high quality data. In future work, we will experiment the effects of data sparsity on more datasets with more similarity computation methods.

References

1. Su, X., Khoshgoftaar, T.M.: A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence 2009, 1–19 (2009)
2. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In: ACM 1994 Conference on Computer Supported Cooperative Work (CSCW 1994), pp. 175–186. ACM, New York (1994)
3. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based Collaborative Filtering Recommendation Algorithms. In: 10th International World Wide Web Conference (WWW 2001), pp. 285–295. ACM, New York (2001)

4. Bobadilla, J., Serradilla, F.: The Effect of Sparsity on Collaborative Filtering Metrics. In: 20th Australasian Database Conference (ADC 2009), pp. 9–18. Australian Computer Society, Inc., Australia (2009)
5. Piccart, B., Struyf, J., Blockeel, H.: Alleviating the Sparsity Problem in Collaborative Filtering by Using an Adapted Distance and A Graph-based Method. In: Tenth SIAM International Conference on Data Mining, pp. 189–198. Society for Industrial and Applied Mathematics, USA (2010)
6. Hofmann, T.: Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis. In: 26th Annual International ACM SIGIR Conference (SIGIR 2003), pp. 259–266. ACM Press, New York (2003)
7. Marlin, B.: Modeling User Rating Profiles for Collaborative Filtering. In: 17th Annual Conference on Neural Information Processing Systems (NIPS 2003), pp. 627–634. MIT Press, Cambridge (2003)
8. Xue, G., Lin, C., Yang, Q., Xi, W., Zeng, H., Yu, Y., Chen, Z.: Scalable Collaborative Filtering Using Cluster-based Smoothing. In: 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005), pp. 114–121. ACM Press, New York (2005)
9. Gu, Q., Zhou, J., Ding, C.: Collaborative Filtering: Weighted Nonnegative Matrix Factorization Incorporating User and Item Graphs. In: 10th SIAM International Conference on Data Mining (SDM 2010), pp. 199–210. Society for Industrial and Applied Mathematics, Philadelphia (2010)
10. Herlocker, J., Konstan, J., Terveen, L., Riedl, J.: Evaluating Collaborative Filtering Recommender Systems. Transactions on Information Systems (TOIS) 22, 5–53 (2004)
11. McNee, S.M., Riedl, J., Konstan, J.A.: Accurate is not always good: How Accuracy Metrics have hurt Recommender Systems. In: 2006 Conference on Human Factors in Computing Systems (CHI 2006), pp. 1–5. ACM Press, New York (2006)
12. Amatriain, X., Lathia, N., Pujol, J.M., Kwak, H., Oliver, N.: The Wisdom of the Few: A collaborative Filtering Approach Based on Expert Opinions from the Web. In: 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009), pp. 532–539. ACM Press, New York (2009)
13. Hu, B., Li, Z., Chao, W., Hu, X., Wang, J.: User Preference Representation Based on Psychometric Models. In: 22nd Australia Database Conference (ADC 2011), pp. 59–66. ACS, Australia (2011)

Data Sparsity: A Key Disadvantage of User-Based Collaborative Filtering?

Biyun Hu , Zhoujun Li, and Wenhan Chao

State Key Laboratory of Software Development Environment, Beihang University, China
`byhu8210@gmail.com, lizj@buaa.edu.cn,`
`cwh2k@163.com`

Abstract. Traditionally, data sparsity is seen as a key disadvantage of user-based CF. It is often assumed that data sparsity may cause small number of co-rated items or no such ones between two users, resulting in unreliable or unavailable similarity information, and further incurring poor recommendation quality. However, the analysis process is often not experimentally verified. To make a detailed analysis, the effects of the data sparsity on user-based CF are experimented with three steps. Firstly, the relationships between the data sparsity and the number of co-rated items are investigated. Secondly, the characteristics of the number are explored. Thirdly, the effects of the number on the recommendation quality are evaluated. Experimental results show that: a) as data sparsity increases, the number of co-rated items doesn't drop, and b) recommendation quality doesn't drop as the number of co-rated items decreases. These results show that the traditional analysis about the effects of data sparsity is problematic. We hope that this new conclusion about the effects of data sparsity can provide implications for the design of CF algorithms.

Keywords: Collaborative Filtering, Data Sparsity, Co-rated Items.

1 Introduction

Traditionally, it is often assumed that user-based CF is susceptible to data sparsity [1, 2, 3, 4, 5], However, when the sparsity problem of user-based CF was proposed by Sarwar et al. [2], no experiments were conducted to verify these analyses about its effects. For the first time, Bobadilla et al. [5] have experimented on the effects. Experimental results manifest that, as the level of data sparsity increases, the recommendation accuracy of user-based CF using Pearson correlation coefficient (ref. Fig. 1 and Fig. 5.a in the paper) even gets better! Their work has shown that: a) data sparsity can have positive effects on the recommendation quality of user-based CF, and b) the traditional analysis about the effects of data sparsity is questionable. To make a detailed analysis of the effects, we first experiment on the relationship between data sparsity and the number of co-rated items, and then explore the characteristics of the number. Next, the relationships between the number and recommendation quality including accuracy and coverage are investigated.

2   Data Sparsity Problem and Related Terms

A typical CF system has a set of users $U = \{u_1, u_2, ..., u_m\}$, a set of items $I = \{i_1, i_2, ..., i_n\}$, and a set of ratings $D = \{r_{u,i} \mid u \in U, i \in I\}$ including user u's rating for item i. In many commercial CF systems, even active users may have purchased (or rated) well under 1% of the items, causing the data sparsity (*ds* for short) problem [2]. Usually, user ratings are converted into a user-item matrix $R_{m \times n}$, and the sparsity level of the matrix is defined as:

$$ds(D,U,I) = 1 - \frac{nonzero\ entries\ in\ R}{total\ entries\ in\ R} = 1 - \frac{|D|}{|U| \times |I|} \tag{1}$$

where $|D|, |U|$, and $|I|$ are respectively the number of ratings, users, and items in a recommendation system.

To make a detailed analysis of the effects of data sparsity, we investigate on each of the key steps in the analysis shown in Fig. 1.
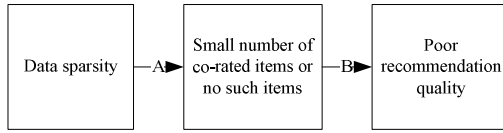
Fig. 1. The traditional analysis about the effects of data sparsity on the recommendation quality of user-based collaborative filtering

Data Sparsity and the Number of Co-rated Items. For the investigation, we divide users' ratings in a system to different subsets $< D_{t_0}, D_{t_1}, D_{t_2}, ... D_{t_n} >$ containing the ratings with different timestamps $< t_0, ... t_i, t_j, ... t_n >$, where $t_0 < t_1 ... < t_n$, $t_1 - t_0 = t_j - t_i$, t_0 is the start time of the system or data collecting, and t_n is the end time. For example, the subset D_{t_1} includes the ratings with the timestamps ranging from t_0 to t_1, and D_{t_n} contains the ratings with those ranging from t_0 to t_n. For each subset of ratings, we then evaluate the level of the data sparsity (ds) and the average number of co-rated items between users (cn) denoted as:

$$cn(D, <U, N>) = avg(\sum_{1 <= i <= m} \sum_{i < j < m} I_{u_i \in U} \cap I_{n_j \in N}) \tag{2}$$

where U is the set of users, N is the set of neighbors for users in U, m is the number of users in U, $I_{u_i \in U}$ is the set of items in dataset D rated by user u_i, and $I_{n_j \in N}$ is the set rated by user u_i's jth neighbor n_j.

The Characteristics of the Number of Co-rated Items. To explore this, in the subset D_{t_n}, for users $U_{D_{t_n}}$ and their neighbors $N_{D_{t_n}}$ with different number of co-rated items, we trace back their number of co-rated items, the average number of ratings for users

$U_{D_{t_n}}$ and their neighbors $N_{D_{t_n}}$ in subsets $<D_{t_1},D_{t_2},...,D_{t_{n-1}}>$ respectively, and investigate the followings: the number of co-rated items at early stage (cn_early), and the generation rate of co-rated items between users (cn_rate):

$$cn\_early = cn(D_{t_1},<U_{D_{t_n}},N_{D_{t_n}}>) \tag{3}$$

$$cn\_rate =$$
$$\frac{(cn(D_{t_n},<U_{D_{t_n}},N_{D_{t_n}}>)-cn(D_{t_1},<U_{D_{t_n}},N_{D_{t_n}}>))}{aur(D_{t_n},U_{D_{t_n}})-aur(D_{t_1},U_{D_{t_n}})+aur(D_{t_n},N_{D_{t_n}})-aur(D_{t_1},N_{D_{t_n}})} \tag{4}$$

where $cn(D_{t_1},<U_{D_{t_n}},N_{Dt_n}>)$ is the corresponding average number of co-rated items for users $U_{D_{t_n}}$ and their neighbors $N_{D_{t_n}}$ in D_{t_1}, and $aur(D_{t_1},U_{D_{t_n}})$ and $aur(D_{t_1},N_{D_{t_n}})$ are respectively the average number of items rated by users $U_{D_{t_n}}$ and their neighbors $N_{D_{t_n}}$ in dataset D_{t_1}. The values of cn_rate is between 0 and 1.

The Relationship between the Number of Co-rated Items and Recommendation Quality. In this step of the investigation, we choose datasets with decreasing number of co-rated items, and explore their recommendation performance.

3   Empirical Analysis

3.1   Dataset and Metrics

The MovieLens dataset [2] is used in the experiments. The dataset contains 100,000 ratings of approximately 1,682 movies made by 943 users, and is collected during the seven-month period from September 19th, 1997 through April 22nd, 1998. Ratings are discrete values from 1 to 5 with recorded timestamps. As Sarwar et al. [2] do, 80% of the dataset was randomly selected into a training set and the remaining into a test set. In our evaluation, one of the most popular used accuracy metrics, the MAE metric is used [2]. It corresponds to the average absolute deviation of predictions to the actual ratings in the test set. Another recommendation quality metric used in this paper is *Coverage* [5]. The coverage is the ratio of predicted ratings to all the ratings N in the test set.

3.2   Procedures

First, divide the training set into 12 subsets $<D_{t_1},D_{t_2},...D_{t_{12}}>$ using the method described in Section 2, and for each subset, compute the values of data sparsity (ds) and the average number of co-rated items (cn); Then choose the subsets with increasing data sparsity, explore the characteristics of the number of co- items (cn), and calculate the values of Equations (3) and (4) in Section 2; Next, choose the subsets with decreasing number of co-rated items (cn), use the subsets respectively and user-based CF based on Pearson correlation coefficient to make predictions for the ratings in the test set, and investigate the relationships between cn and

recommendation quality including accuracy and coverage. The neighborhood size of user-based CF is set to 5, 10, … , and 60 respectively, and average results for these settings are reported.

3.3 Results

Data Sparsity and the Number of Co-rated Items. The levels of data sparsity (ds), and the average number of co-rated items (cn) for each subset of training ratings $<D_{t_1}, D_{t_2}, ... D_{t_{12}}>$ are reported in Figs. 2 and 3. Fig. 2 shows, as time passes, the data sparsity level of ratings tends to increase. This manifests that the increment of ratings is slower than that of users and items. We can see from Fig. 3 that, the number of co-rated items between users tends to increase over time. We infer that this happens because the number of co-rated items increases accordingly when users use recommendation systems more and more ratings are produced.
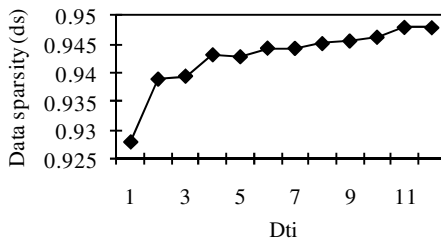
Fig. 2. The levels of data sparsity for each subset of training ratings D_{t_i} with increasing timestamps t_i
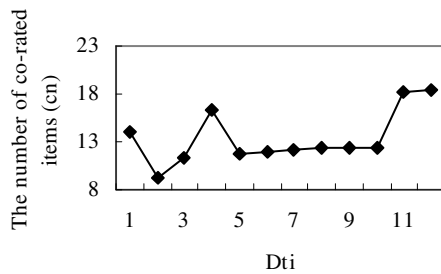
Fig. 3. The average number of co-rated items for each subset of training ratings D_{t_i} with increasing timestamps t_i

When comparing Figs. 2 and 3, we see that, although as time passes, data sparsity increases, the number of co-rated items doesn't drop but almost ascends. This is contrary to the step A of the analysis shown in Fig. 1.

The Characteristics of the Number of Co-rated Items. The subsets $<D_{t_2}, D_{t_3}, D_{t_5}, D_{t_6} ..., D_{t_{10}}>$ are chosen for the following experiments, because for these

subsets, data sparsity keeps increasing. In the subset $D_{t_{10}}$, for users $U_{D_{t_{10}}}$ and their neighbors $N_{D_{t_{10}}}$ with different numbers of co-rated items, the corresponding numbers of co-rated items in other subsets and the values of co-rate speed (cn_rate) are reported in Fig. 4. The figure shows that, there are mainly two categories of users and neighbors:

Category 1: Users and neighbors who co-rate more at early stage ($cn\_early = cn(D_{t_2}, <U_{D_{t_{10}}}, N_{D_{t_{10}}}>)$ in Fig. 4.a is larger), and co-rate more quickly as they stay longer in the system (cn_rate in Fig 4.b is also larger).
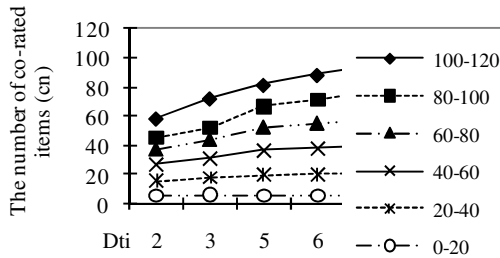
(a) Their corresponding number of co-rated items in D_{t_i}
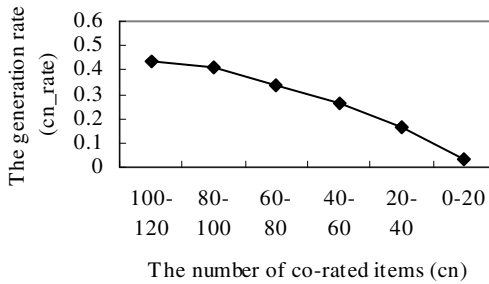
(b) The generation rate of these co-rated items

Fig. 4. For users and neighbors with different numbers of co-rated items in $D_{t_{10}}$

Category 2: Users and neighbors who co-rate fewer at early stage, and co-rate much slower as time passes.

It can also be seen from Fig. 4 that, Category 1 includes the users and neighbors with high numbers of co-rated items and those with low numbers can be attributed to Category 2. For example, for users and neighbors with the numbers of co-rated items ranging from 100 to 120, these users and neighbors are in Category 1. They co-rate 59 items at early stage (Fig. 4.a), and the co-rate speed of them is 0.43 (Fig. 4.b). The values of cn_early and cn_rate for these users and neighbors are much larger than the values for those users and neighbors with the numbers of co-rated items ranging from 0 to 20 ($cn\_early = 5$, $cn\_rate = 0.03$).

The number of co-rated items and user similarity: In summary, these results show that, for users in Category 1, they co-rate more at first, and they tend to co-rate even more over time, while for users in Category 2, the number of co-rated items between them almost does not ascend, and they manifest this at early stage by few number of co-rated items. It is intuitive to think that the users and neighbors in Category 1 are more similar with each one; while those in Category 2 are less similar.

The Relationship between the Number of Co-rated Items and Recommendation Quality. The results for investigating the relationships between the number of co-rated items and recommendation accuracy and coverage are reported in Fig. 5, which shows, MAE decreases and Coverage increases as the number of co-rated items decreases (for the subsets $< D_{t_{10}}, D_{t_9}, ...D_{t_5}, D_{t_3}, D_{t_2} >$, the number is decreasing, ref. Fig. 3), i.e., recommendation accuracy and coverage increase as the number of co-rated items drops. This means that the step B of the traditional inference shown in Fig. 1 is not experimentally supported (contrary results shown).
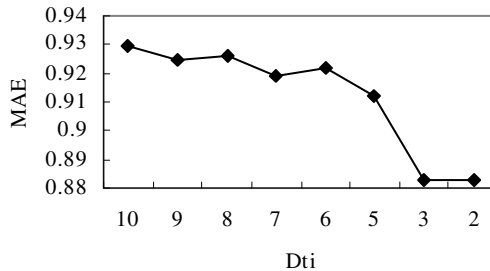
(a) MAE (keeps dropping)
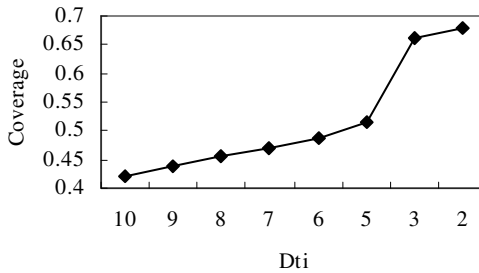
(b) Coverage (keeps increasing)

Fig. 5. Recommendation performance of subsets $< D_{t_{10}}, D_{t_9}, ...D_{t_5}, D_{t_3}, D_{t_2} >$ with decreasing number of co-rated items

4   Discussion

Experimental results show that neither the step A nor the step B shown in Fig. 1 is supported, while contrary results for the two steps are manifested. Trends shown by

experimental results are, as time goes by, data sparsity becomes more severe (ref. Fig. 2), the number of co-rated items increases (ref. Fig. 3 or the increment trends in Fig. 4.a), and recommendation quality including accuracy and coverage drops (ref. Fig. 5, in reverse order of D_{t_i}). This series of results is shown in Fig. 6. While the step A in the figure are easy to understand, it is not directly explained why more co-rated items result in poor recommendation accuracy and coverage. We mainly explain this in the followings.
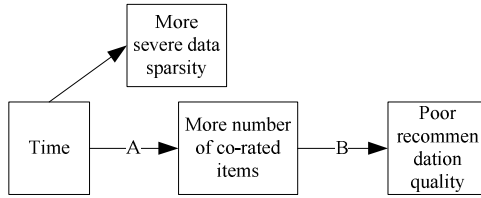
Fig. 6. The effects of user rating sparsity on the recommendation quality of user-based CF

Co-rated Items and Recommendation Accuracy. Recommendation accuracy is mainly related to the reliability of user similarity, so we can infer that more co-rated items cause less similar users be chosen as neighbors. To verify this, on the subsets $< D_{t_{10}}, D_{t_9}, ...D_{t_5}, D_{t_3}, D_{t_2} >$, for users U_D and their neighbors N_D with different number of co-rated items, we compute the average similarity between them, and the average results for these subsets are shown in Fig. 7, which manifests that, as the number of co-rated items increases, the similarity drops. This pushes neighbors with less number of co-rated items be chosen (we call this phenomenon as neighbor drift), however, when judging by the co-rate trend between users, these neighbors are in Category 2 and are less similar as analyzed in Section 3, we think that this causes poor recommendation accuracy when users co-rate more.
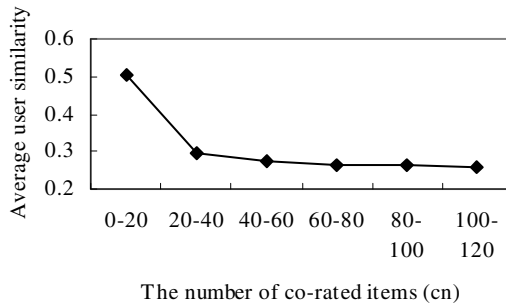
Fig. 7. Average similarity values for users and neighbors with different number of co-rated items

Co-rated Items and Recommendation Coverage. The analysis above shows that as users co-rate more, the neighbors with fewer co-rated items will be chosen, it follows that recommendation coverage drops because these neighbors are less useful in predicting active users' preferences comparing with those having more co-rated items.

5   Conclusion

Experimental results in this paper show, as time passes, data sparsity tends to become more severe, but it has no negative effects on the number of co-rated items, which increases or keeps low for more similar users or less similar ones. It is also shown that small number of co-rated items doesn't cause poor recommendation accuracy and coverage, in contrary; the two quality metrics are improved. Further analysis shows that, when users co-rate more, neighbors with less co-rated items are pushed to be chosen, while these neighbors are less similar when analyzing from the co-rate trend between users, and are less useful for users, causing poor recommendation accuracy and coverage. In all, the trends shown by experimental results are: as time passes, the number of co-rated items between users increases, neighbors with less co-rated items are pushed to be chosen, causing poor recommendation accuracy and coverage.

References

[1] Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 230–237. ACM, New York (1999)

[2] Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295. ACM, New York (2001)

[3] Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. on Knowl. and Data Eng. 17(6), 734–749 (2005)

[4] Grčar, M., Mladenič, D., Fortuna, B., Grobelnik, M.: Data Sparsity Issues in the Collaborative Filtering Framework. In: Nasraoui, O., Zaïane, O.R., Spiliopoulou, M., Mobasher, B., Masand, B., Yu, P.S. (eds.) WebKDD 2005. LNCS (LNAI), vol. 4198, pp. 58–76. Springer, Heidelberg (2006)

[5] Bobadilla, J., Serradilla, F.: The effect of sparsity on collaborative filtering metrics. In: Proceedings of the Twentieth Australasian Conference on Australasian Database, vol. 92, pp. 9–18. Australian Computer Society, Inc., Darlinghurst (2009)

Path Skyline for Moving Objects

Wookey Lee[1], Chris Soo-Hyun Eom[1], and Tae-Chang Jo[2]

[1] Department of Industrial Engineering
[2] Mathematics,
Inha University, South Korea
{trinity,sunching,taechang}@inha.ac.kr

Abstract. Skyline query has been used mainly for relatively static and low dimensional data sets. We develop the Skyline query for the moving objects coping with dynamic changes efficiently. This study is focused on deriving a fundamental algorithm for extracting the path skylines so that the Shortest Path based algorithm, named PathSL, can generate an optimal skyline for moving objects. It turns out that PathSL is robust against changing the source and destination and generically scalable for the problem size with polynomial computational complexity.

Keywords: Path Skyline, Moving objects, Shortest path.

1   Introduction

1.1   Basic Issues on Skyline

Skyline query has been known as one of the effective and efficient methods to deal with the large amounts and multi-dimensional data. By employing the notion of 'dominance', the skyline query can isolate the target Skyline data so that the dominated ones can effectively be debarred as avoidable data. Conventional research issues have been mainly dedicated to relatively static data sets. Recently the data collection mechanism have been developed and their utilization necessities have been rushed tremendously, and therefore the data becomes more and more complex, explosive, and dynamic such as GIS, Web & Social Network, UCC, Data Warehousing and Business Intelligence, Patent Information, Sensor Network, Criminal and Surveillance, Bio and Medical, Worldwide Marketing, Stock Market, Culture Industries, Satellite Images, Smartphone Apps, etc. [2, 14, 19, 22].

The term, skyline, is originated from the resemblance of the contour line of a city silhouette so that it can quickly figure out the big picture of the whole city. By utilizing the concept of 'dominance', skyline data can be separated from dominated data which one can discard for a skyline query. Skyline represents formally the set of points which dominate the other points with at least one attribute value. Using the

skyline, users can be helped for the decision making with respect to the dimensions of interests and dynamic environment changes.

1.2   Motivational Example

Assume a road network as Fig. 1. Given the source node 1 and the destination 6, consider the following questions. What is the shortest path for this? The answer will be simple: "1–2–6" path. Note that even though the map size increased a lot, there are $O(n\ log\ n)$ algorithms for this issue [4,18]. What are the distances of each and every node for that shortest path? Specifically, the question changes as what is the distance from each node (e.g., node 3) to that shortest path? It depends on the definition of the distance. The alternatives result in (1) Euclidian distance (e.g., 30) [18], (2) Path or Manhattan distance (e.g., 25) [8], (3) Minimum distance from the spot perpendicular to the node (e.g., 5) [9, 12, 15], or shortest path approach [22]. Assume that each node represents gas-station and the car will move from node 6 to node 5. Which gas-station is the nearest and cheapest alternative? That question forms *the path skyline problem* in this paper.
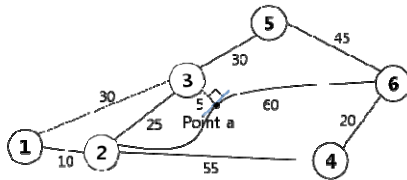
Fig. 1. Road Network

Another consideration comes from the dynamic situation changes. In the middle of driving, what if the destination will be changed from node1 to node6, and then what are the distances? It is inevitable for the conventional skyline methods to recalculate all the paths. Therefore, the attribute changes can trigger the cases that cannot be responded fast enough due to the high computation burden. We want to deal with the *path skyline* problem and that can be solved effectively for the dynamic changes. To reduce the re-computation affliction of skyline derivation for the moving objects, we present a novel approach suggesting an optimal skyline by generating the distance formulation for the shortest paths. In our approach, the nodes including source and the destination are allowed to change, so that the corresponding new skylines can efficiently be extracted with relatively small amount of computation load.

The rest of this paper is organized as follows. Section 2 presents our path skyline algorithm for moving objects, while Section 3 presents some results of comparing our algorithm and the experimental results with discussion. Section 4 described the related works and finally, Section 5 presents the conclusions for our research.

2   Path Skyline Algorithm for the Moving Objects

Recently, applications which support Location-Based Services (LBS) offer some supplementary services like automobile navigation systems or traffic reports by locating some moving objects or individuals. Actually, to use LBS for mobile applications, efficient searches for the location information of the moving objects and static objects (e.g., the gas station or expressway rest area) are required. Also, a shortest path needs to be derived.

We consider the property that the Euclidian distance is always shorter than or equal to the network distance [17]. Dijkstra algorithm [4] and A* algorithm [9] have been the representative methods to get a shortest path. Even adopting these algorithms to find the shortest path, the skyline method makes still better to deal with various and more complicated situations.

One of the most advantageous properties of our algorithm is the measuring distance of each node on the path. Previous algorithms have calculated a skyline based on the distance between query position and each node, thus whenever the query position changes, the distance and the skyline should be recalculated. It is an unbearable limitation for the moving and ever changing objects. We established a strategy to estimate the distance of each node as the shortest distance between the query position and the destination via the path node. Therefore, it is needless for the moving objects to recalculate both the distance and skyline.

The brief description of the Path Skyline algorithm is as follows. It consists of two phases: The first one is to find the shortest path from the source n_s to each and every node and then to find the shortest path from each node to the destination node n_t. The second one is to find the Skyline for the path. That is formally described as a theorem 1, but we speaking, the distance for node n is estimated by the distance from n_s to n_t via n. Namely, the shortest distance of n_s-n and n-n_t are saved in *previous*[n_j], and the distance for node n is saved in $d[n_j]$.

In this paper, the data set is indexed into R-tree (even if our approach is not limited to a certain index structure), and all the objects in the root node of the R-tree are sorted by the minimum distance calculated according to the theorem. The minimum distance for each object is measured by the distance between the zero point and the left lower point of the object. The objects newly inserted are expanded in terms of the descending order of minimum distance, and the expansion process is performed by deleting the object and inserting child objects of the parent object. The process is repeated until leaf nodes are inserted. When the leaf node is reached, the leaf node is then inserted into the skyline list L if only the node is not dominated by any node of L. In summary, the object expansion, deleting, and dominance estimation are repeated until the list L is completed. The complexity of our approach is $O[n/2\ log\ n/2 + n/2\ log\ n/2] = O[n\ log\ n]$, since we adopted shortest path search algorithm twice.

Algorithm: **PathSL**

> **Input: G(N, E), n$_s$, n$_t$**
> **Output: Skyline nodes**

G(N,E); N: the given set of data nodes = {n$_1$, n$_2$, …}
> **E: the given set of edges = {(n$_i$, n$_j$)} for n$_i$, n$_j$ ∈ N;**
> **weight(n$_i$, n$_j$): weight of edge(n$_i$, n$_j$); n$_s$: start node;**
> **n$_t$: target node, where n$_s$, n$_t$ ∈ N and n$_s$≠n$_t$;**
Begin
> **N*: N\{n$_s$, n$_t$}; n ∈ N*; previous[n] = ø;**
> > **If n ∈ N* is null**
> > **n = ShortestPath(G, n$_s$, n$_t$);**
> > **while n ∈ N* is not null and |n| > 0**
> > > **distance of n: ShortestPath(G, n$_s$, n) + ShortestPath(G, n, n$_t$)**
> > > **N*: N* \{n};**
> > **end while**
> **L ← Result**
> **Generate R-tree T from N**
> **L = ø**
> **// list of skyline points**
> **While heap ≠ ø**
> > **remove top object o**
> > **if remove o when it is dominated**
> > **else**
> > > **// o is not dominated**
> > > **If o is not a data point**
> > > > **for each child o$_i$ of o**
> > > > > **expand o and insert o$_i$ ∈ o for o$_i$ is not dominated**
> > > > **end for**
> > > **else**
> > > **// o is a data point**
> > **L ← o**
> **End while**
End.

Fig. 2. Path Skyline (PathSL) Algorithm

3   Experimental Results and Discussions

Experiments are performed in Intel Xeon 5130 2.00 GHz (4 CPUs) with 4090MB RAM and Windows Server 2003 Enterprise operation system. Experimental data for synthetic set consists of the data such that the number of nodes varied 200, 400, 600, 800 and 1000, and the data point has up to 30 dimensions including *distance* attribute. Given each distance for the path in Fig. 3, this illustrates at first that the shortest path between the query position "S" and the destination node "T" is "S-2-6-T" that has total distance 100 that is represented by bold lines. In this paper, the distance determination for each and every node to the shortest path is different from conventional approaches. For example, our approach (PathSL) gives that the path via node 3 is, as discussed in the above, 105, since the shortest distance between the start node S and

node 3 is 35, and from which the shortest distance to node T is 80. Regarding the conventional approach the distance for the node 3 is 150, since the round trip distance 50 (node 2 and 3) is added to the shortest path 100. All the others can be derived in the same way.
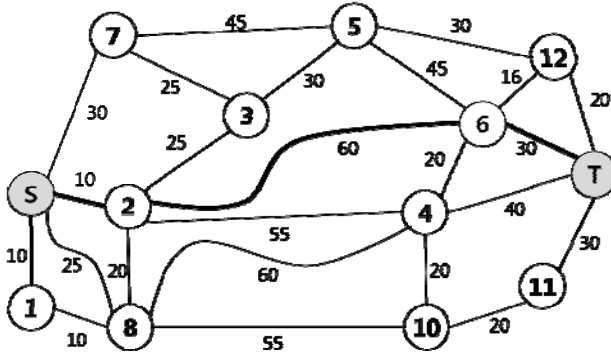
Fig. 3. An example for the Path SL and the shortest path
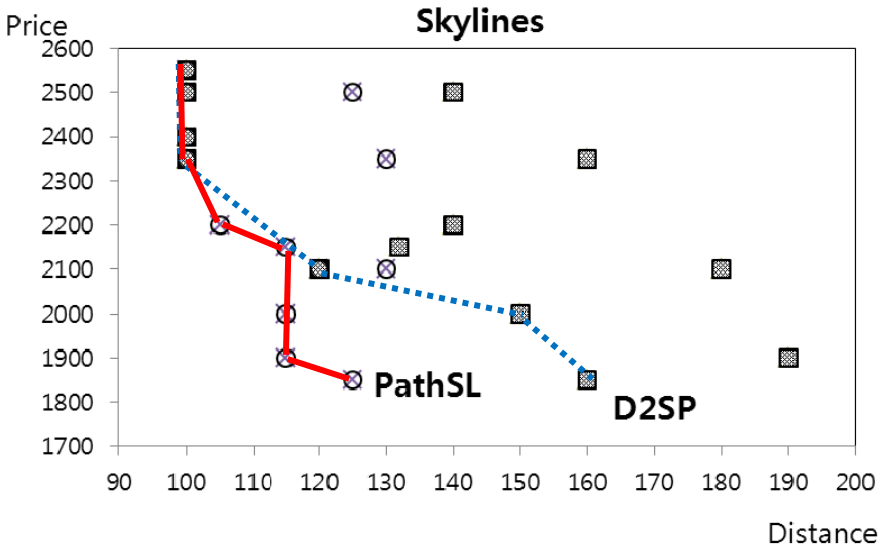
Fig. 4. Skylines by PathSL and the conventional approach D2SP

In figure 4, there are the result skylines by our PathSL and the conventional approach D2SP (named, from the distance to the shortest path). First of all on the same example, most of the distances by PathSL are reduced by the algorithm where the distances on the nodes of the shortest path "S-2-6-T" are the same, 100. On deriving Skyline, at first, the static Skyline points are node 2, 1, and 7. The PathSL, however, shows upgraded results: node 2, 4, 5, and 7 where the node 4 and 5 are added and

node 1 is deleted from the skyline. Since the distance of node 4 is changed from 140 to 105, and the distance of node 5 is changed from 190 to 115, respectively. The new results are not only reduced values but also the enhanced paths in the new skyline.

In this paper, as the object is moving, recalculation of the distance should be required when a query position or the destination node are changed. When the source node and/or the destination node will be changed, re-computation must be inevitably followed. It can be achieved for our approach more efficiently, compared with those for the previous approaches that the entire recalculation should be required. We overcome the unnecessary computation burden tremendously, but that is omitted due to the size limitation.

We compared the performance of our PathSL algorithm with the frequently used algorithm, BBS based approach. The following Fig. 5 represents the experimental results. The left-hand side figure represents the comparison on the distance and the right-hand-side figure the location change, respectively. Notice that the robustness for the dynamic changes can be found in the right-hand side figure so that PathSL can consume about half time for the original PathSL algorithm. The larger the number of the data set, the better our algorithm effects, so that the PathSL definitely outperforms to the BBS based approach.

(a)   (b)
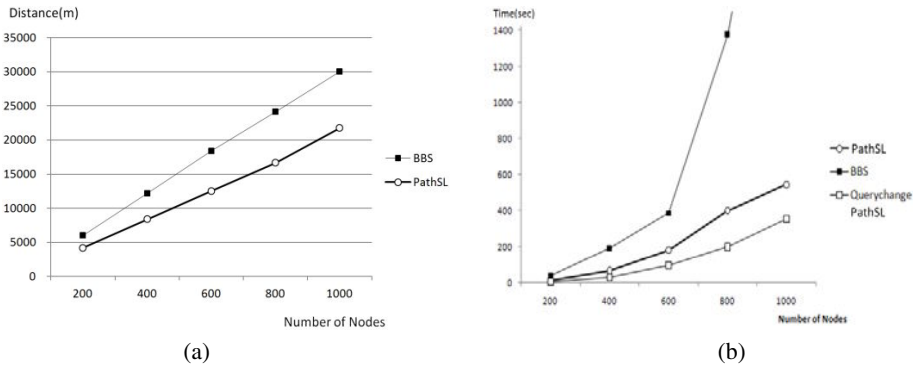
Fig. 5. Comparison of Algorithm Performance

4   Related Works

Research for the Skyline has been pouring in these days, but seldom has been on *the path skyline*. Fundamental skyline concepts and tools have been appeared by Divide and Conquer [2], Block Nested Loop [19], Nearest Neighbor Search [7], Branch and Bounded Search [5], XML Querying [3], Distance-Based Skyline [20], etc. Various environments have been tackled for Uncertain Data [1], and Multi-Relational Operator [13], Stochastic approach [6, 16], Categorical data [15], etc. Conventional skyline extraction methods have been limited for the low dimensional dataset has been efficiently performed by existing methods like SUBSKY [21] that users do not consider all of the dimensions, but only several dimensions according to their intent and personalities.

Focusing on the moving object, [21] presents an algorithm that computes skyline after efficient recalculation of varying distances as the object moves. [12] and [23] suggested a clustering approach for the moving object and a comprehensive index mechanism. The recalculation method, however, remains unsolved, when the query position changes rapidly. Additionally, recent applications like Information Retrieval system and Geographical Information System should deal with the high dimensional dataset. Such a high dimension requires an exponential computation time that faces inevitably the curse of dimensionalities. To overcome the problem, [24] used Object based Partitioning algorithm, which efficiently allocates the sliced data into parallel storage and the algorithm has a defect in that it is not software solution but uses hardware performance. We, however, develop a polynomial and robust algorithm even for the dynamic target and source changes. Apparently, it is not a complete solution. Detailed skyline issues should be brief here for the space limitation.

5   Conclusions

In the large multi-dimensional and dynamic moving object database, the conventional methods for extracting skyline have mainly been developed on the lower dimensional and static data set. However, as new field like GIS with mobile devices generated applications, Web & Social Network, DW & BI, Patent Information, Sensor Network, Surveillance, Bio and Medical, Marketing, Stock Market, Culture Industries, Satellite Images, etc. becoming more and more complex and dynamic, so we developed an algorithm for *path skyline* for those kinds of data efficiently. Also, previous research cannot be effectively applied into the moving objects like car, airplane, robot, satellite, etc. with initial and destination point. Since the conventional skyline approaches for continuously moving object have been an optimal only for the current static position and they are not guaranteed the optimal for the entire moving path and extremely fragile for the dynamic changes. In this paper, we propose PathSL (Path SkyLine) algorithm using the Shortest Path with the Path skyline computation algorithm. We verified the performance of our algorithm by many of the experimental results. The enhanced results by PathSL are not only reduced values but also the dissimilar paths in the new skyline. Specifically, we showed that the PathSL can find the optimal skyline for the entire moving path, and is generically scalable for the problem size with polynomial computational complexity.

References

1. Atallah, M.J., Qi, Y.: Computing All Skyline Probabilities for Uncertain Data. In: Proc. ACM PODS, pp. 279–787 (2009)
2. Börzsönyi, S., Kossmann, D., Stocker, K.: The Skyline Operator. In: Proc. ICDE, pp. 421–430 (2001)

3. Cohen, S., Shiloach, M.: Flexible XML Querying Using Skyline Semantics. In: Proc. ICDE, pp. 553–564 (2009)
4. Dahl, O.J., Dijkstra, E.W.: Hoare Structured Programming. Academic Press, London (1972)
5. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An Optimal and Progressive Algorithm for Skyline Queries. In: Proc. SIGMOD, pp. 467–478 (2003)
6. Sacharidis, D., Arvanitis, A., Sellis, T.: Probabilistic Contextual Skylines. In: Proc. ICDE, pp. 273–284 (2010)
7. Kossmann, D., Ramsak, F., Rost, S.: Shooting Star in the Sky: An Online Algorithm for Skyline Queries. In: Proc. VLDB, pp. 275–286 (2002)
8. Samet, H., Sankaranarayanan, J., Alborzi, H.: Scalable Network Distance Browsing in Spatial Databases. In: Proc. SIGMOD, pp. 43–54 (2008)
9. Hart, P.E., Nilsson, N.J., Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths in Graphs. IEEE TSMC 4(2), 100–107 (1968)
10. Hsueh, Y., Zimmermann, R., Ku, W., Jin, Y.: SkyEngine: Efficient Skyline Search Engine for Continuous Skyline Computations. In: Proc. ICDE, pp. 1316–1319 (2011)
11. Köhler, H., Yang, J.: Computing Large Skylines over Few Dimensions: The Curse of Anti-Correlation. In: Proc. APWeb, pp. 284–290 (2010)
12. Jensen, C.S., Lin, D., Ooi, B.C.: Continuous Clustering of Moving Objects. IEEE TKDE 19(9), 1161–1174 (2007)
13. Jin, W., Ester, M., Hu, Z., Han, J.: The Multi-Relational Skyline Operator. In: Proc. ICDE, pp. 1276–1280 (2007)
14. Lee, W., Leung, C.K., Lee, J.J.: Mobile Web Navigation in Digital Ecosystems Using Rooted Directed Trees. IEEE TIE 58(6), 2154–2162 (2011)
15. Lee, W., Song, J.J., Leung, C.K.-S.: Categorical Data Skyline Using Classification Tree. In: Du, X., Fan, W., Wang, J., Peng, Z., Sharaf, M.A. (eds.) APWeb 2011. LNCS, vol. 6612, pp. 181–187. Springer, Heidelberg (2011)
16. Lin, X., Zhang, Y., Zhang, W., Cheema, M.A.: Stochastic Skyline Operator. In: Proc. ICDE, pp. 721–732 (2011)
17. Papadias, D., Zhang, J., Mamoulis, N., Tao, Y.: Query Processing in Spatial Network Databases. In: Proc. VLDB, pp. 802–813 (2003)
18. Sharifzadeh, M., Shahabi, C., Kazemi, L.: Processing Spatial Skyline Queries in Both Vector Spaces and Spatial Network Databases. ACM TODS 34(3), 1–45 (2009)
19. Tan, K.L., Eng, P.K., Ooi, B.C.: Efficient Progressive Skyline Computation. In: Proc. VLDB, pp. 301–310 (2001)
20. Tao, Y., Ding, L., Lin, X., Pei, J.: Distance-Based Representative Skyline. In: Proc. ICDE, pp. 892–903 (2009)
21. Tian, L., Wang, L., Zou, P., Jia, Y., Li, A.: Continuous Monitoring of Skyline Query over Highly Dynamic Moving Objects. In: Proc. MobiDE, pp. 59–66 (2007)
22. Yoon, S., Ye, W., Heidemann, J.S., Littlefield, B., Shahabi, C.: SWATS: Wireless Sensor Networks for Steamflood and Waterflood Pipeline Monitoring. IEEE Network 25(1), 50–56 (2011)
23. Zhang, M., Chen, S., Jensen, C.S., Ooi, B.C., Zhang, Z.: Effectively Indexing Uncertain Moving Objects for Predictive Queries. In: Proc. PVLDB, vol. 2(1), pp. 1198–1209 (2009)
24. Zhang, S., Mamoulis, N., Cheung, D.W.: Scalable Skyline Computation Using Object-Based Space Partitioning. In: Proc. SIGMOD, pp. 483–494 (2009)

A Graphical Audit Facility for Data Processing and Its Evaluation with Users

Jens Müller, Murat Kavak, and Klemens Böhm

Information Systems Group,
Institute for Program Structures and Data Organization,
Faculty of Informatics, Karlsruhe Institute of Technology (KIT), Germany
{jens.mueller,klemens.boehm}@kit.edu, murat.kavak@student.kit.edu

Abstract. Personally-identifiable information (PII) is increasingly processed in a distributed way. This makes it much harder for individuals to oversee how their PII is used. In the legal systems of many countries, processing of PII is subject to restrictions. In particular, companies have to inform an individual on how they use his PII, and which external parties they transfer it to. We hypothesize that naïve approaches like log messages or plain text are not sufficient to this end. We in turn have developed a user-friendly auditing facility based on business processes (BPs). It visualizes data processing in real time, using the graphical process models one would deploy on a BP engine for execution. We also propose an approach to let a BP-management system generate the necessary audit events at runtime. An evaluation of realistic scenarios with users shows that our tool helps them to understand how their PII is used.

1 Introduction

Today, companies outsource parts of their processes to other companies that can perform them more efficiently. As an example, think of a loan-approval process: Consumer loans are highly standardized products with low profit margins. Banks and other organizations granting loans share information about risk factors and credit defaults through specialized credit bureaus like SCHUFA in Germany. As part of the loan approval, the creditor queries risk information from such an agency. In case of a credit default, the agency is informed. This example shows that outsourcing leads to personally-identifiable information (PII) to be transferred to third parties. PII is protected by law in the European Union [3] and elsewhere. The law gives individuals whom the data relates to the right to request information on how their data is processed, and where it is transferred. It also requires the informed consent of individuals to any processing and transfer of PII that is not necessary for the service provided. In order to give this *informed* consent, the user must be able to assess how his or her data will be used.

Current mechanisms fulfill the law formally, but are not useful in reality. Companies usually require individuals to give consent by signing terms and conditions that are both very broad and detailed. Information on data processing that companies provide usually is in text form. With large amounts of text, it is hard for users to find the details they are interested in. When PII is processed in

a distributed way, companies have to tell individuals which other companies they
have transferred it to, or where they have acquired it. In principle, this allows
users to ask the other companies for information on their data. In practice, this
is too tedious and time-consuming for users, especially when they have to follow
their data over multiple hops. We conclude that information must be structured
in a way that is easily accessible to individuals.

Business-process management (BPM) supports the complete lifecycle of or-
chestrations, i. e., applications combining lower-level functionality, from models
to executable processes. Non-functional requirements, including security require-
ments, can be expressed as annotations to graphical process models [6]. Using
an aspect-oriented approach, they can be translated into a process running in
a BPM system (BPMS) with a standard business-process engine extended with
security components.

The goal of this article is to study how to let users track business processes
that use and transfer their data with ease. Because the right to information is not
limited to finished cases, users must be able to get information about running
processes as well. This is known as *auditing*. A BPMS should automatically
generate the necessary events at run-time. To design auditing tools, we also need
to understand how real users work with them. All these tasks are challenging,
for various reasons at different levels:

- We need a succinct representation of audit information that is easy to under-
  stand at first glance. A respective system should also allow for drill-down in
  order to get more details. In particular, it should be easy to switch to another
  process following the data flow. Such a representation is not obvious.
- It is not possible to determine which auditing features are useful for users
  without a realistic scenario they themselves are part of.
- The functionality envisioned should re-use artifacts that are created anyway
  when modeling and deploying a business process. This minimizes the ad-
  ditional effort for application developers. This point concerns two issues in
  particular: First, audit information must be presented to the user. This step
  can re-use graphical business-process diagrams. However, additional infor-
  mation about their structure is needed. Second, the audit tool must acquire
  the necessary information at runtime. This requires an interface between the
  audit tool and the security components of the BPMS. However, the imple-
  mentation is not obvious, as we will explain.

To this end, we have designed and implemented an auditing tool dubbed WoSec
(*Wo*rkflow *Sec*urity) and have evaluated it with real users. More specifically, our
contributions are as follows:

- We have analyzed which information user need in order to audit data transfer
  in distributed applications, and how it can be presented visually.
- We have developed WoSec, a web-based tool for auditing the handling of PII
  based on graphical BPMN models. It allows users to "follow their data" when
  it is transferred to another organization that also provides data to WoSec.
  It can also be used to visualize how an organization *intends* to handle PII,
  enabling users to give more informed consent.

- We propose an extension of a BPMS with extensions for managing security configuration and enforcing security decisions (secure BPMS) to provide events to the audit tool.
- We have designed several example use cases for distributed data processing that are sufficiently complex for a realistic evaluation.
- We have designed a user study for evaluating our tool and various features of it. Having carried out the study, it shows that users prefer graphical audit facilities, and that these lead to a better understanding of data transfers. We have discovered that usability is curbed severely when process diagrams do not fit the screen without scrolling. This finding as well as other ones give way to an improved version of our audit facility.

2   Fundamentals

Because our auditing approach is based on BP models, we introduce some fundamentals regarding (secure) BPM in this section. BPM is an ideal solution for distributed information processing using loosely coupled systems. It is used to orchestrate (i.e., coordinate) the behavior of different components and actors. The starting point is a BP diagram, which is typically graphical, where a domain expert explicitly models control and data flow.

The Workflow Management Coalition has proposed a reference architecture of workflow-management systems (WfMS), the workflow reference model [5], which still fits BPMS in service-oriented architectures [4]. Interface 5 concerns *Audit and Monitoring.* The WfMC has specified a format for audit messages [10]. However, this format is focused on states of process and activity instances and does not address the handling of data items and the relationship between different processes through message exchanges.
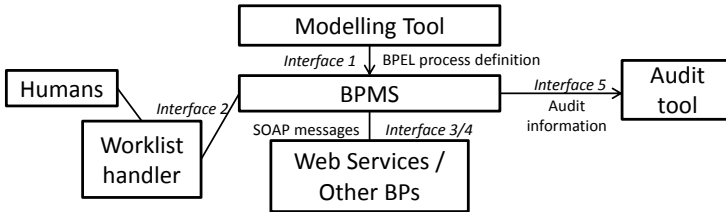
Fig. 1. Interfaces of the WfMC workflow reference model

There exists a proposal for an architecture of a secure BPMS that takes context into account and allows respective configuration [7]. This architecture extends a conventional BPEL engine with security components. These security components capture and store the context of BP instances. This context, such as who has performed activities, is important for audit in the general case. The components also enforce security decisions regarding interactions of the BP with the outside world. BPs are instrumented using an AOP-style approach in order to run in that extended BPMS.

3 Functionality

In the following, we present requirements on the functionality of our audit tool from the user perspective[1]. We have derived them by inspecting systematically which kinds of data that users are interested in arise in real-world processes. In more detail, we take a user-centric approach: We think of business processes handling data of one individual. This individual is allowed to track how his data is processed. To this end, he can access a list of BP instances handling his data and a detailed audit view for each of them. This view contains historical and live information, as we will explain below.

General Structure: Because BPMN is a generally accepted standard, and BPMN diagrams already exist for applications modeled as BPs, we decide to use them as the basis for the visualization. Note that such diagrams are static and do not contain information on the current state of process instances. They contain lanes (horizontal) representing the process (coordinating the overall application behavior), roles involved in it, and external parties, activities (rectangles), solid arrows (mostly horizontal) representing the control flow, and arrows (mostly vertical, dashed) representing the data flow between the process and persons/external parties.

Execution Progress: WoSec visualizes execution progress by highlighting activities already executed and currently executing in this diagram. When an activity starts execution, the tool automatically scrolls the viewport to that activity and notifies the user acoustically. Detailed information about activities is available, such as which user performed it at which time.

Data Transfer: A data object moving between activities indicates Data transfer (Fig. 2a). An information box provides details about data transferred (Fig. 2b). The audit view allows access to historical information as follows: A timeslider allows to move to some point in the past and cause a playback of events from there. In case of loops, activities can be executed more than once. Information boxes for activities also contain information about earlier executions of an activity, not only the most recent one.

Multiple Diagrams: When multiple organizations are involved in a process, several diagrams show the perspective of each participating organization, where the processes of other organizations are only represented by their interfaces called by the current process. This leads to smaller diagrams that are easier to oversee. It must be easy for users to discover what happens with their data in the other process. WoSec accomplishes this as follows: When an activity transfers data to another BP, the user can jump to the other BP instance from there. WoSec then opens it in another view and scrolls to the activity that receives the data.

Color Scheme: See [8] for an detailed explanation of the scheme used.

---

[1] See [2] for a live demo of WoSec.

(a) Animation of data flow

(b) Information box with details about an activity
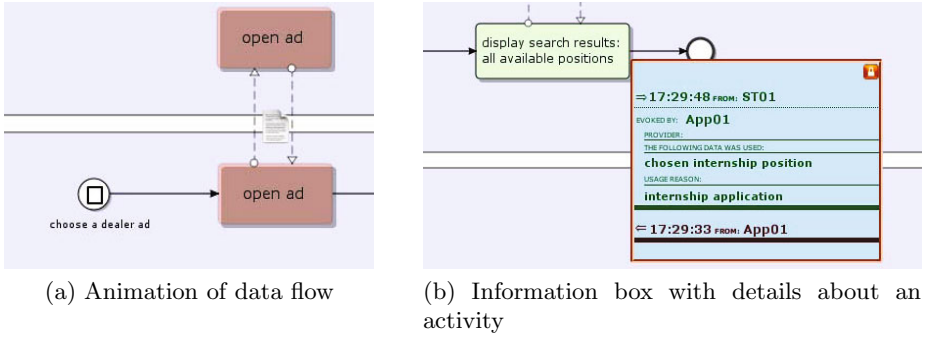
Fig. 2. Screenshots of WoSec

4   Architecture and Design

We now describe the architecture of our application that yields the functionality presented in Section 3: **Providing audit information:** BP definitions are instrumented, i. e., modified so that they generate audit events, similar to [11]. **Creation of graphical process models:** Domain experts and process modelers initially create business processes as BPMN diagrams in a graphical modeling tool. These models are translated into executable BPEL processes. From the same BPMN diagrams, we need to create graphical representations for our auditing tool. Intalio BPMN Designer produces a SVG version of process diagrams where elements are annotated with activity names. The SVG graphic and an additional description file are then deployed to WoSec. Fig. 3 shows how the auditing tool fits in the overall BPMS architecture. **Internal Architecture of WoSec:** WoSec itself displays audit information by overlaying it on the SVG diagrams. We have chosen a client/server design because we need a server part that stores BP definitions and is able to receive audit events at any time. We implement this as a web application because this is easy to use and modern browsers natively support SVG.
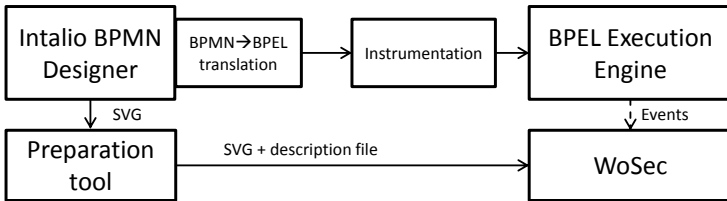
Fig. 3. Integration of the auditing tool into the overall BPMS architecture

5   Study Design

The goal of our study is to find out whether our visualization improves the understanding of users of how their PII is processed. In addition, we want to discover shortcomings of WoSec, possible misconceptions of how audit information should be presented to users, and consequently opportunities for improving WoSec in particular and audit facilities for end users in general. To achieve these goals, we need an appropriate scenario and a realistic baseline for comparison. We also need to measure the understanding of the participants. This requires asking participants for their assessment according to criteria we deem important. In addition, we need to check these answers for plausibility. We do this by asking questions about what has happened in the scenarios and analyzing the behavior of the participants during the study. We use textual audit messages as the baseline, as organizations still use it to answer information requests and to fulfill their legal obligations. The scenarios used in the study must be sufficiently complex. This means that several organizations should be involved in processing different kinds of data. The scenarios must match the interests of the study group. Although they are less detailed than in the real world, they must appear as natural as possible. We provide a realistic web application and the corresponding audit view at the same time, using a split browser window. The upper half shows the mock-up of a web application, while the lower half contains an audit view, which either uses our visualization or is text-based. We chose to evaluate an early prototype of WoSec in order to get preliminary feedback. This pre-study revealed some shortcomings in the implementation leading to visualization errors. The (quite young) participant did not fully understand the purpose of such an audit system, but clearly preferred the graphical version regarding user-friendliness (mean: 5.69) and clarity (mean: 5.13). [8] contains a more thorough description of the pre-study.

5.1   Main Study

For the main study, we used an improved version of the visualization with more features. We have tested the following hypotheses:

- H1: Our visualization helps users to understand which of their data is transmitted to which organizations.
- H2: The visual audit facility has good usability.
- H3: Users prefer a graphical audit facility over a text-based one.

Because the participants of our pre-study have not been able to really see the benefits of an audit facility, presumably because of their young age, we chose to perform the main study with more experienced participants. To evaluate the functionality of WoSec in full, we devised a more sophisticated study setup. We decided to use two different configurations of the visualization, in addition to the

text-based audit view: While both configurations allow to access all audit information available, one contains additional features aiming at improved usability. This allows to test the influence of the non-essential features and to explore the opinion of the participants regarding the additional functions. To avoid learning and order effects when comparing the configurations, we used different scenarios for the different configurations. The participants went through the scenarios in the same order, but the order of configurations was randomized. In both configurations, the visualization was immediately updated when something relevant had happened in the web application. Both contained the basic animations, i. e. activities starting execution blink, and data transfer is animated. In addition, the full-featured configuration automatically scrolls to activities becoming active. It also shows the actual data transferred, the user causing the activity, and the reason for transferring the data. For activities, a window shown on right-click contains a textual description of the activity and all users involved. We recruited our study participants from our directory of individuals interested in user studies related to information systems, which includes mostly university students and adults with university education. We designed three scenarios specifically for this group: (1) *Internship application:* In this scenario, eligible students get support for placement in an internship program, which relates to their course of study. First, participants enter their registration information. They then wait for several steps: A university coordinator has to approve their application, registration data is written to a database, the university coordinator chooses a placement service, which then sends a list of possible internships. Finally, the participant chooses an internship. (2) *Online trade:* In this scenario, participants have to sell an item. First, they have to enter their trader data and the item description. They then wait for another customer to open the ad, buy the item and send his or her contact data. The participant now gets the address of the buyer, has to prepare the parcel and hand it to a parcel service. (3) *Car purchase:* Here, the participants have to buy a car on credit. They have to wait for a list of available cars and choose one. They then have to enter their personal data and, in our example, state that they want to buy the car on credit. They choose a bank for the loan and accept the terms of the credit bureau. They then have to wait for several steps happening in the background: The bank receives the credit application and receives a score from the credit bureau. We assume that it grants the application. The car dealer receives a confirmation from the bank, reserves the car chosen and sends a purchase confirmation to the customer. We paid the participants 10 EUR for their participation. To incentivize active participation, we promised an additional amount based on the level of participation. We computed this amount based on the number of questions answered. This means that participants who answered all 60 questions were paid another 5 EUR. In addition to the questions answered, we recorded for which scenarios, tasks, and participants information boxes for activities were shown. To get an overall impression how the user interface was used, we recorded a so-called heatmap, overlaying mouse clicks onto a screenshot of the user interface.

6 Results

In total, 17 individuals have participated in our study. The study group included participants of different age (20–74 years). All of them had some technical background, and all expressed some privacy concerns regarding Internet usage. We performed the study in two separate meetings with participants, with 7 of them in the first and 10 in the second one. Due to technical problems, we were only able to test the full visualization at the first meeting. This means that 7 participants used the full visualization for all three scenarios. We weighted them with $1/3$ when computing the mean values to achieve the same weight per participant. We compared the ratings given by the participants regarding the text-based and the graphical audit view, as well as the ones for the restricted and the full graphical version. We first tested all samples for normal distribution with the Shapiro/Wilk test [9]. Because this test did not confirm a normal distribution, we had to use the Wilcoxon/Mann/Whitney test to compare samples. The average answers and the result of the significance tests are shown in Table 1. For all tests, we required a level of significance $\alpha = 0.05$. We performed one-sided tests whether the underlying random variable of the sample with the larger mean actually is significantly larger. Table 2 contains the mean values of the questions asking for a direct comparison. The samples have not shown a significant difference from the neutral value 4. In Section 5, we have hypothesized that our visualization helps users to understand which of their data is transmitted to which organizations (H1). The statistically significant difference for Q1, Q2, Q3, Q8, and Q9 shows that the visualization indeed led to a better understanding of data transfers. Regarding H2: *The visual audit has good usability*, we cannot show any statistical significance for Q5, Q6, Q7 and Q12. This indicates that the

Table 1. Assessment of different audit configurations

	G	T	F	R	G/T^s	R/F^s
Q1 Have you been able to trace the flow of the data?[S1]	5.71	4	5.31	6.1	>	=
Q2 Have you been able to trace why a data flow has happened?[S1]	5.99	4.3	5.78	6.2	>	=
Q3 Could you predict following steps?[S1]	5.43	4	5.26	5.6	>	=
Q4 How was the number of animations?[S2]	4.8	3	4.6	5	>	=
Q5 Have you been able to orient yourself without problems?[S1]	5.01	5.2	4.51	5.5	=	=
Q6 How clear was the auditing tool?[S3]	4.61	4.1	3.92	5.3	=	>
Q7 How user-friendly was the auditing-tool?[S4]	4.6	4.3	4.39	4.8	=	=
Q8 How much information content did the auditing tool contain?[S5]	5.34	3.9	4.63	6.13	>	>
Q9 Do you feel adequately informed about all actions?[S1]	4.93	3.9	4.75	5.11	>	=

Legend:
G : all graphical versions   T : textual version F : full graphical version
R : restricted graphical version s : significance test
S1 : 1 = absolutely not, 7 = absolutely yes   S2 : 1 = too few, 7 = too much
S3 : 1 = not clear at all, 7 = absolutely clear S4 : 1 = not user-friendly at all, 7 = absolutely user-friendly
S5 : 1 = very few information content, 7 = very much

Table 2. Direct comparison of text-based and graphical audit

	Mean
Q10 How clear was the textual representation compared to the graphical one?[S1]	4.8
Q11 How much information did the textual representation contain compared to the graphical one?[S2]	3.5
Q12 How user-friendly was the textual representation compared to the graphical one?[S3]	3.3
Q13 Which visualization do you prefer?[S4]	4.5

Legend:
S1 : 1 = little clear, 7 = very clear   S2 : 1 = very few, 7 = very much
S3 : 1 = not user-friendly at all, 7 = very user-friendly S4 : 1 = rather textual, 7 = rather graphical

usability of all audit systems is equal. Nevertheless, the restricted visual audit scores significantly better than the full one. We conclude that automatic changes of the viewport decrease usability. As one may expect, we could not show any statistically significant difference for H3, *Users prefer a graphical audit facility over a text-based one* through questions Q10, Q11, Q12 and Q13. Yet, except for a small outlier (Q10), the participants evaluated the visual audit slightly better than the textual audit. In total, our tool improves the understanding of users, but there is potential for better usability. In particular, participants were annoyed by automatic scrolling. On the other hand, some participants have noted that the size available for the diagram was too small, and the heatmaps showed that the scrollbars were used a lot. We conclude that automated scrolling should be improved, or scrolling should be made unnecessary. We believe that splitting diagrams into parts and easy navigation between the parts can alleviate the respective problems. Participants mostly accessed the additional information provided by the information boxes when answering the questions. This indicates that they had used the audit facility mainly for that purpose. In a future study, participants should answer the control questions solely based on their usage of the system up to that point.

7   Related Work

The WfMC has specified a audit data format for business processes [10] and started the development of an XML-based successor [12]. Both formats focus on the execution only, i.e., the state of BP instances and activities. They do not address data handling and the messages exchanged between BP instances. The *BPMS Console* of [1] lists available BP definitions and instances and marks activities currently running. However, it does not address data transfer and is targetted at administrators, not end users. To the best of our knowledge, the effectiveness of generic audit facilities for data processing dedicated to end users has not been studied or empirically evaluated in the literature.

8   Conclusions and Future Work

We have created a graphical tool that allows end users to audit business processes involving data transfer and have described its integration into a BPMS. We have carried out a study comparing the tool with text-based audit facilities, which represent the current state of the art, and have assessed its impact on effectiveness and usability. Next to other points, the results show a usability problem related to limited viewport sizes and automatic scrolling. As future work, we plan to assess whether splitting diagrams into parts can alleviate these problems. We also plan to automate the BPMS integration.

References

1. Intalio BPMS Designer, `http://www.intalio.com/bpms/designer`
2. WoSec website, `http://dbis.ipd.uni-karlsruhe.de/english/1746.php`
3. European Community: Directive 95/46/EC (Data Protection Directive)
4. Hollingsworth, D.: Workflow Handbook 2004, vol. 10, ch. The Workflow Reference Model 10 Years On (2004)
5. Hollingsworth, D.: The Workflow Reference Model. WfMC Specification TC00-1003, Workflow Management Coalition (1995)
6. Mülle, J., von Stackelberg, S., Böhm, K.: Modelling and Transforming Security Constraints in Privacy-Aware Business Processes. In: Proc. SOCA 2011 (2011)
7. Müller, J., Böhm, K.: The Architecture of a Secure Business-Process-Management System in Service-Oriented Environments. In: ECOWS 2011 (2011)
8. Müller, J., Kavak, M., Böhm, K.: A Graphical Audit Facility for Data Processing and its Evaluation with Users. Tech. Rep. 2012-1, Karlsruhe Reports in Informatics
9. Shapiro, S.S., Wilk, M.B.: An analysis of variance test for normality (complete samples). Biometrika 3(52), 1–22 (1965)
10. Workflow Management Coalition: Audit Data Specification (1998)
11. Yao, J., Chen, S., Wang, C., Levy, D., Zic, J.: Accountability as a service for the cloud. In: SCC 2010 (2010)
12. zur Muehlen, M. (ed.): Business Process Analytics Format (BPAF). WfMC Draft Standard WFMC-TC-1015 (February 2008)

---

[2] The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

Efficient SPARQL Query Processing in MapReduce through Data Partitioning and Indexing

Zhi Nie[1,2], Fang Du[1,2], Yueguo Chen[1], Xiaoyong Du[1,2], and Linhao Xu[3]

[1] Key Laboratory of Data Engineering and Knowledge Engineering,
Ministry of Education, China
[2] School of Information, Renmin University of China, Beijing, China
{niezhi,dfang,chenyueguo,duyong}@ruc.edu.cn
[3] IBM Research China, Beijing, China
xulinhao@cn.ibm.com

Abstract. Processing SPARQL queries on single node is obviously not scalable, considering the rapid growth of RDF knowledge bases. This calls for scalable solutions of SPARQL query processing over Web-scale RDF data. There have been attempts for applying SPARQL query processing techniques in MapReduce environments. However, no study has been conducted on finding optimal partitioning and indexing schemes for distributing RDF data in MapReduce. In this paper, we investigate RDF data partitioning technique that provides effective indexing schemes to support efficient SPARQL query processing in MapReduce. Our extensive experiments over a huge real-life RDF dataset show the performance of the proposed partitioning and indexing schemes for efficient SPARQL query processing.

Keywords: SPARQL, MapReduce, data partition, schema identification, JAQL.

1 Introduction

The Resource Description Framework (RDF) [1] data model has been widely used for describing semantic web data [3], which rapidly grows with the advances of information extraction and natural language processing techniques. RDF model organizes the Semantic Web data in a set of triples. Each triple (also called a statement) is of the form (subject, predicate, object). The SPARQL query language for RDF is an official standard for searching over RDF repositories [2]. It defines constraints over subjects, predicates (which can also be called properties) and objects, so that only triples satisfying the constraints are retrieved. At present, the RDF data is growing rapidly. The Semantic Web community has addressed a Billion Triple Challenge [3] that consists of more than 3.2 billion triples. From the perspective of relational databases, triple stores have been designed to manage the RDF data and query the data with SPARQL. The most famous are Sesame [21], Jena [23] and so on. However, they are all designed on the single node with the supports of existing relational database techniques; hence they have limited ability on scalability.

MapReduce [4] is introduced by Google to perform parallel computations on very large datasets. The distributed file system proposed as Google File System (GFS) [5] provides functionality to store a large file over multiple storage nodes by partitioning and replicating. Hadoop [6] is an open source implementation of the MapReduce and distributed file system (HDFS) similar to GFS. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.

To achieve parallel data processing for SPARQL queries over the MapReduce framework, we need to take the recent achievements of Hadoop for handling massive scale Web data on clusters. MapReduce is a low level framework which is not often convenient for managing and querying structured data. To facilitate the operations of large scale data over the MapReduce framework, some high level languages such as JAQL [7], PigLatin [8], SCOPE [9], and DryadLINQ [10] have been proposed. In Hadoop framework, a file is the smallest unit of input to a MapReduce job and always read from the disk. It will be better if RDF data can be effectively partitioned and indexed so that the whole dataset is not necessary to be loaded and scanned. In this paper, we study the performance of three RDF partitioning strategies when they are applied for distributing RDF data in MapReduce. We present a solution to transform the SPARQL query to JAQL operators, using JAQL query language on top of Hadoop to search RDF data of JSON format.

The main contributions of our work can be summarized as follows. Firstly, we propose a solution for SPARQL queries in MapReduce. Secondly, we design an algorithm to transform SPARQL queries into JAQL. Finally, we study the performance of different RDF partitioning strategies in MapReduce environment. The experimental results over large real-life dataset demonstrate that the significant performance improvements achieved by the proposed partitioning and indexing mechanisms.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 introduces the architecture of our Hadoop based parallel processing framework and gives detailed information on the transformation of SPARQL to JAQL. Section 4 presents data partitioning and indexing strategies. Section 5 shows the experimental results and we conclude in Section 6.

2   Related Work

Early works, such as Sesame [21] and Jena [23], use relational tables to store RDF triples. Another approach to store RDF triples is using column-store [22]. In column store, triples are divided into predicate tables and each table corresponds to a file whose name represents the predicate. Other studies, RDF-3X [19] and YARS2 [20] store six different combinations of triples as indices. RDF-3X is widely accepted as the state of art for SPARQL query processing engine on single node. But when the data grow up to billions, the index scan becomes a bottleneck.

Paper [13] provides theoretical designs of a parallel processing framework for RDF data. Urbani [18] shows us a way of doing scalable distributed reasoning over RDF dataset. Yahoo Research group [14] proposed a way to map the SPARQL algebra to Pig by rewriting SPARQL queries in Pig representation. In [15], the authors present a method extending the Pig to process RDF data on top of the Hadoop infrastructure. However, Pig engine does not consider the characteristics of RDF data and

implement multi-way join. SPIDER [16] is a demonstration of how to store and query graphs using HBase. Authors of [17] propose to split large RDF data into more small files according to the predicate. These files are named with predicate and then stored using Hadoop's HDFS. The subject and objects are stored in these predicate files. SPARQL queries are broken down into subqueries that can be processed using MapReduce jobs. In this paper, we focus on different partitioning mechanism that can efficiently support SPARQL query processing in MapReduce.

3   Architecture Overview

Based on the JAQL language [7], we design and implement a system that is able to process SPARQL queries in the MapReduce framework. The architecture of our system is illustrated in Figure 1. JSON [12] is used as the basic data model for recording the RDF data. The key appeal of JAQL is its ability to automatically generate a query plan of MapReduce jobs from a high level language and further optimize those jobs. SPARQL transformer translates SPARQL query to JAQL operators, which are then conducted in the MapReduce environment.
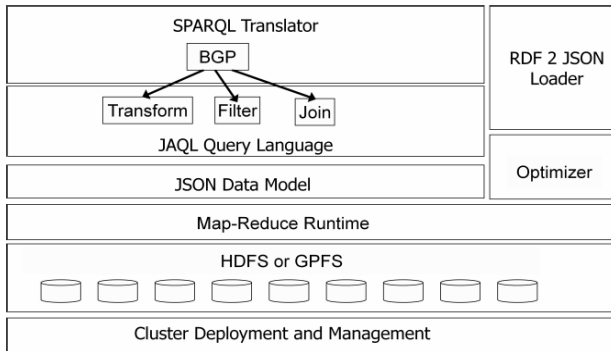
Fig. 1. RDF data query processing architecture

3.1   JSON and JAQL

JSON [12] is a lightweight data-interchange format. It consists of two complex types (arrays and records) and four atomic types (number, string, boolean, and null). In our framework, each dataset is represented in JSON by an array (delimited by brackets) of objects (delimited by braces). E.g., the RDF triples are represented as the array [{s: KunMing, p: long, o: 102°E},{s: KunMing, p: lat, o: 25°N}, …], where the fields: s, p and o, correspond to subject, predicate and object respectively; long and lat correspond to the longitude and latitude features. We can identify the values of triples according to the additional labels of s, p and o. JAQL [7] is an open-source language for querying JSON data. It is built from several core expressions that are designed to operate on large arrays through parallelization. They are called JAQL operators such as filter, transform, join.

3.2 SPARQL to JAQL Transformation

In this paper, we simply consider the conjunctive pattern translation. We explain the details of transformation in Table 1. The **read** operator is used to load data from file stored in distributed file system HDFS. The label "$" is an implicitly defined variable that binding to the array value, the "." used in $.s uses to point label value. The **filter** expression filters away elements from its input array. The **transform** expression projects each element of its input array. The **join** expression is used to express a join between two or more input arrays. The join condition between two inputs is assumed to be an equi-join. **Algorithm 1** shows the transformation from SPARQL to JAQL. The *GetCorrespondFiles* will return different results for different partitioning strategies. Assuming that RDF triples are partitioned using column-store (i.e., partition triples by predicates), the results of *GetCorrespondFiles* are predicate files corresponding to the predicate names specified in query BGPs.

Table 1. SPARQL to JAQL

SPARQL Query	PREFIXpos:<http://www.w3.org/2003/01/geo/wgs84_pos#>, foaf:http://xmlns.com/foaf/0.1/name> SELECT ?lat ?long WHERE {?a foaf:name "KunMing". ?a pos:lat ?lat. ?a pos:long ?long."}

JAQL Query	//read file from hdfs by predicate name
	    $1 = **read**(hdfs('foaf:name'))
	       -> **filter** $.o == "KunMing"
	       -> **transform** {$.s}; //select and projection
	    $2 = **read**(hdfs('pos:lat'));
	    $3 = **read**(hdfs('pos:long'));
	   join $1, $2, $3 **where** $1.s == $2.s and $2.s == $3.s
	   into { lat:$2.o, long:$3.o }; // project to ?lat ?long

4 Data Partitioning and Indexing

By transforming SPARQL queries into JAQL operators, we are able to run SPARQL queries in MapReduce. However, the generated JAQL operators are highly dependent on how data are partitioned and distributed. One straightforward partitioning strategy is to store all the data in one file, which means that the Hadoop system need to scan the entire data in the read operation. Although loading dataset can be parallelized by multiple mappers, obviously, it is wasteful to load and process a large percentage of data that are not relevant to the query. In this section, we introduce how data partitioning methods can help to solve this problem in the MapReduce framework.

Horizontal Partitioning: In this strategy, RDF data are partitioned into files by subjects. All the triples having the common hash value of subject are partitioned to the same file. Then we create indices for predicates and objects in RDF triples. For queries, given the value of a subject, the corresponding triples can be quickly located. However, when the subject is a variable, horizontal partitioning will be not efficient for retrieving the relevant triples. As described in section 3.2, **Algorithm 1** gives the

method to transform SPARQL to JAQL, if the subject in the query triple is a variable, instead of scanning all files, the *GetCorrespondFiles* function will get the files based on indices and the result of this function will be the intersection of these files.

Algorithm 1. Transformation of SPARQL to JAQL

Input : SPARQL query: q

Output : JAQL query: q'

LVa   = null;   // a list to getting literals in each query triple $t_i \subset q$

LVar = null;   // a list to getting variable in each query triple $t_i \subset q$

JoinVar = null;   //hashmap to getting the same variable name

1. Parse the q and get the variable set s_k in select clause and t_i in where clause

2. **For** each $t_i \subset q$ **do**

3.   **if** $t_i.s$ is variable **then**   //t_i: triple *i* in *where* clause, s :subject

4.   LVar .add ($t_i.s$); JoinVar.map($t_i.s$);   //the common variable will map together

5.   **else** LVa.add($t_i.s$);

6.   **if** $t_i.p$ is variable **then**   // p: predicate

7.   LVar.add($t_i.p$); JoinVar.map ($t_i.p$);

8.   **else** LVa.add($t_i.p$);

9.   **if** $t_i.o$ is variable **then**   // o: object

10.   LVar.add($t_i.o$); JoinVar.map ($t_i.o$);

11.   **else** LVa.add($t_i.o$);

12. FileList fl = GetCorrespondFiles(LVa); // function input: LVa

13. **For** each $f_i \subset fl$ **do**

14.   q'.add(**read** f_i + **filter** LVa$_i$);   //add **read** and **filter** operator

15. **For** each $v_i \subset$ JoinVar **do**

16.   **If** count(v_i) > 0   q'.add(**join** + JoinVar.get(v_i)); //add **join** operator

17. q'.add(**transform** +s_k);   //project to select variables

18. return q';

Vertical Partitioning: According to predicates, we divide triples and assign all triples of the same predicate to a file (named by its predicate). Given the value of a predicate, the query can quick find the relevant file. However, the size of files will be not even processed by this method, as some files are much larger than others. The cost is very expensive if all large predicate files are involved. The function *GetCorrespondFiles* will directly return the files corresponding to the queried predicate.

Clustered Property Partitioning: In the third approach, RDF triples are firstly organized as entities. An entity is a RDF tuple with same subject (e.g. a group of all RDF triples with same subject). The definition of an entity is described in our work [11]. Then the entities are partitioned into clusters. In the paper, we also call predicate sets in an entity as the schema of the entity. Through partition, entities with similar schema are clustered into same cluster. When SPARQL queries come, the approach can forward the queries to relevant clusters instead of whole data sets. The clustered property partition outperforms the former two approaches in most cases. However, when all the objects in query are variable (which is rarely happen), or the relevant cluster has very big size, it still need much time to scan the cluster file. Thus we give another

ways to prune the scan space for good performance. After clustering, the partition can also be processed by horizontal and vertical partition strategies. When processing a query, it gets the corresponding cluster partition, and then uses the same method of horizontal or vertical partitioning to load the data files.

5   Experiments

A Hadoop cluster is used for our experiments that has 37 nodes. One node is a master, and the others are slaves. All nodes of the cluster run Hadoop 0.20.2 under Ubuntu 10.04 linux 2.6.32-24-server 64bit. Each node has 47G memory, 4.3TB disk space and 24 processor of Intel(R) Xeon(R) CPU E5645@ 2.40GHz. We use Java 6 to implement our algorithm and run our experiment. The version of JAQL [7] is 0.5.1. Except the replication is 2, we use the default configuration of Hadoop to run our experiments. We used the dataset of Billion Triples Challenge 2010 (BTC10) [29]. It contains data from sources, including Yago, DBpedia, Freebase and others. It is an RDF graph with 3.2B <s, p, o, q> quads. By removing the quad field and deduplicating, we use dictionary encoding to compressed BTC10 from 624 GBs to 50 GBs. The resulting dataset has 1,426,823,976 unique triples; 159,185,030 unique subjects; 95,345 unique predicates and 253.741,625 unique objects. We store the data in HDFS [6] with JSON [12] data model. The JSON is stored as binary format, so it would not impact the performance of query processing.
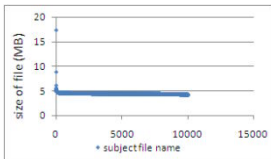
Fig. 2. Data distribution of Horizontal partition
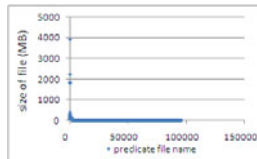
Fig. 3. Data distribution of Vertical partition
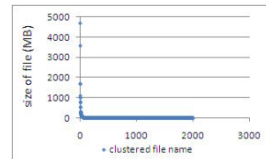
Fig. 4. Data distribution of Clustered Property partition

As described in section 4, we use three methods to partition the BTC10 dataset. Most of files have the same size in Figure 2. Figure 3 shows the distribution of vertical partitioning. As we can see, there are some files very big than the others and the biggest one is the predicate *rdf:type* in the dataset. So when a query contains such predicate, it will cost much time on scanning the file. Figure 4 shows the distribution of the size of clustered property tables. Beside this, we repartition the clustered property file by horizontal and vertical methods.

We use 10 queries to compare the performance of our partition strategies. They are listed in *http://iir.ruc.edu.cn/~niezhi/*. These queries involve four categories. Q3, Q10 are star join queries with poplar predicates and unspecified object. Q 1, Q4, Q5, Q6, Q8, Q9 are also star join but with one or more known object. Q2 is a chain query and Q7 gives the value of subject. The results are shown in figure 5.

Horizontal partition is the worst one for the queries, except Q7. Because when there are unspecified subjects in query (it is the common case), Although the query can use

index to reduce the files, it still need lots of scan operations when predicate or object are partitioned into many partition files. Q7 gives the value of subject, so it just uses hash function to find the candidates. For Q8, vertical partitioning is more expensive than the cluster-based approaches because the query contains a popular predicate of *rdf:type*. Q2, Q5 and Q9 are expensive because the queries have more join operator than the others. The cluster partition strategy is better than vertical partition in Q4, Q6, and Q8 to Q10. With popular predicates or specified objects in star queries, the strategy can minimize the index scan space. However, it is more expensive with Q1, Q2, Q3 and Q5 in that when there are unpopular predicates with unknown objects, the approach also needs to load a large portion of indexes to locate the results. The clustered-vertical partition has the best performance, simply because it takes the advantages of cluster partitioning and vertical partitioning strategies. The cluster-horizontal partition does not make any progress compared to clustered partitioning strategy, except for Q8 (for Q8 give the value of subject). The experimental results basically show that the clustered-based partition strategy achieves the best performance in most cases.
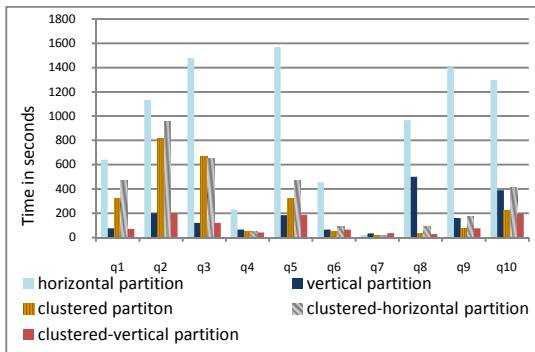
Fig. 5. Cost time of each query

6   Conclusion

In this paper, we exploit a novel approach for parallel data processing over the MapReduce framework and the distributed file system. It is the first time that we use JAQL to process SPARQL on Hadoop and use JSON format to record RDF data. We study different partitioning and indexing strategies for different query types and the clustered property partitioning strategy shows a good performance.

References

1. Resource Description Framework, http://www.w3.org/RDF/
2. SPARQL query language for RDF,
   http://www.w3.org/TR/rdf-sparql-query/

3. Semantic web challenge, `http://challenge.semanticweb.org`
4. Jeffery, D., Sanjay, G.: MapReduce: Simplified data processing on large clusters. In: 6th Conference on Operating System Design and Implementation (2004)
5. Ghemawat, S., Gobioff, H., Leung, S.T.: The Google file system. In: 19th ACM Symposium on Operating Systems Principles, pp. 29–43 (2003)
6. `http://hadoop.apache.org/`
7. JAQL, `http://code.google.com/p/jaql/`
8. Olston, C., Reed, B., Srivastava, U., Kumar, R., Tomkins, A.: Pig Latin: a not-so-foreign language for data processing. In: ACM SIGMOD (2008)
9. Chaiken, R., Jenkins, B., Larson, P., Ramsey, B., Shakib, D., Weaver, S., Zhou, J.: SCOPE: easy and efficient parallel processing of massive data sets. In: PVLDB (2008)
10. Yu, Y., Isard, M., Fetterly, D., Badiu, M., Erlingsson, U., Gunda, P.K., Currey, J.: Dryad-LINQ: A system or general purpose distributed data parallel computing using a high-level language. In: OSDI (2008)
11. Fang, D., Yueguo, C., Xiaoyong, D.: Partitioned Indexes for Entity Search Over RDF Knowledge Bases. In: 17th International Conference on Database Systems for Advanced Applications (2012)
12. JSON, `http://www.json.org`
13. Paolo, C., Andy, S., Chris, D.: A parallel processing framework for RDF design and issues. Technical report, HP Laboratories (2009)
14. Peter, M., Giovanni, T.: Web semantics in the clouds. Yahoo Research (2009)
15. Tanimura, Y., Matono, A., Lynden, S., Kojima, I.: Extensions to the Pig data processing platform for scalable RDF data processing using Hadoop. In: Data Engineering Workshops (ICDEW) (2010)
16. Hyun-sik, C., Jihoon, S., YongHyun, C., Min, K.S., Yon, D.C.: SPIDER: a system for scalable, parallel/distributed evaluation of large-scale RDF data. In: 18th ACM Conference on Information and Knowledge Management, pp. 2087–2088 (2009)
17. Husain, M.F., Khan, L., Kantarcioglu, M., Thuraisingham, B.: Data Intensive Query Processing for Large RDF Graphs Using Cloud Computing Tools. In: 3rd IEEE International Conference on Cloud Computing (2010)
18. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable Distributed Reasoning Using MapReduce. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 634–649. Springer, Heidelberg (2009)
19. Thomas, N., Gerhard, W.: Rdf-3x: a risc-style engine for rdf. In: PVLDB, vol. 1(1) (2008)
20. Harth, A., Umbrich, J., Hogan, A., Decker, S.: YARS2: A Federated Repository for Querying Graph Structured Data from the Web. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 211–224. Springer, Heidelberg (2007)
21. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 54–68. Springer, Heidelberg (2002)
22. Daniel, J.A., Adam, M., Samuel, R.M., Kate, H.: Scalable semantic web data management using vertical partitioning. In: VLDB (2007)
23. Wilkinson, K., Sayers, C., Kuno, H.A., Reynolds, D.: Efficient RDF Storage and Retrieval in Jena2. In: 1st International Workshop on Semantic Web and Databases (2003)

An Entity Class Model Based Correlated Query Path Selection Method in Multiple Domains

Jing Shan, Derong Shen, Tiezheng Nie, Yue Kou, and Ge Yu

Institude of Computer Software and Theory, College of Information Science and Engineering,
Northeastern University, Shenyang, 110004, China
shanjing@research.neu.edu.cn
{shenderong,nietiezheng,kouyue,yuge}@ise.neu.edu.cn

Abstract. One interesting problem in data integration is, when users submit a query in one domain, he may also want to know more information in other domains which haven't come up to his or her mind yet. It's a great idea to recommend correlated domains to users so that they could get more information. However, the existing researches in solving this problem are quite rare and limited. In this paper, we propose a method of selecting query paths, and the query paths consist of correlated domains. We propose an entity class model which is based to construct a domain correlation graph, and query paths are picked based on the graph. At last we evaluate our method by experiments and demonstrate the effectiveness of our method.

Keywords: entity class model, multiple domains, correlation graph, query path.

1 Introduction

Compare to information hidden in Web-accessible databases (which we call Deep Web), information contained in surface web is only a small part of the overall information available on the Web. Therefore, information integration in Deep Web has been identified as the next big challenge.

The existing researches mainly focus on integrating information which comes from data sources in the same domain, for example, integrating book information from different bookshops. However, when a user submits a query in one domain, it doesn't mean he wants to get information just from that domain, the user may also want to know more information in other domains which hasn't come up to his mind yet. Information seeking can be seen as a process-intensive task. A user often starts from a vague information need and progressively discovers more on his need, with a mix of look-up, browsing, analysis and exploration. So, in this paper, we are interested in helping user to discover domains correlated to his initial query need, and recommend query paths to him. For example, a user makes a leisure trip plan and wants to search for a travel attraction at first, then he may also interest in a concert close to the attraction, considering also close-by hotels and traffic routes. All the information the user needs is from four domains, and a query path like "travel attractions→concert→hotel→traffic" can be constructed according to the query intention. Given an initial query domain, finding its correlated domains and constructing query paths is the research goal of this paper.

There are two key problems, one is how to discover correlations between domains; the other is how to select query paths based on those correlations. We propose an entity class model to describe characteristics of each domain, which is not limited by heterogeneous data sources. Then we propose a method to discover the correlations between domains based on entity class model, and construct a correlation graph. According to the graph we pick out query paths using Markov Chain Algorithm.

The overall contributions of this paper are: (i) An entity class model to describe characteristics of domains. (ii) A method to calculate correlations between domains. (iii) A method to select query paths out of the correlation graph based on Markov Chain Algorithm. (iv) Evaluations on large sets of data sources.

2   Related Work

Our research integrates data across multiple domains. In the existing researches, Mashup is somehow similar to our purpose. Mashups[1] integrate a set of complementary Web services and data sources, to construct a new service. There is a system called COMPASS[2] which assists users in their navigation through MashAPPs. NGS[3] is a framework providing fully automated support for cross-domain queries over multiple, specialized search engines. Besides, there are also some Web services oriented cross-domain integration systems. For example, Gang Chen et al [4] proposed a cross-service travel engine for trip planning, and Alessandro Bozzon et al [5] proposed a search computing framework for multi-domain queries. The former mainly faces the travel domain, integrates travel services and resources based on a geographical ontology; the latter faces multiple domains, registers data sources as services, decomposes the query into sub-queries and maps each sub-query to a domain-specific search service, and builds query results by combining the outputs produced by service calls.

The work most similar to our research is proposed by Yingjun Li et al [6]. That approach focuses on finding the correlation between domains and recommending top-k cross-domain query paths to users. However, there are some deficiencies and limitations in that approach. First, when mining correlations, the approach mainly depends on attribute information of advanced query interfaces, and the limitation is, if data sources don't provide advanced query interfaces, the approach cannot be used. Second, when recommending top-k query paths, the approach uses a multi-factor fuzzy evaluation model [7] to select query paths. However, that model can only pick out paths whose length is 1, that is, just one pair of domains, and there is only one data source in each domain. That is not enough for data integration. Considering these deficiencies, we propose an entity class model based query path selection method.

3   Framework and Entity Class Model

3.1   Framework

We assume that all the data sources have been classified by domains, and each domain has a global query interface and a global query result schema. It's an important foundation for the entity class model which is about to be introduced. The framework

is shown in Figure 1. When a user sends an initial query, he needs to select the global query interface of the initial domain, and then the node corresponding to the domain is located in the correlation graph. After that, we find correlated domains and form query paths. The construction of correlation graph is finished in the preprocessing phase. The correlation graph is a directed weighted graph, nodes represent domains, and each domain contains a set of data sources.
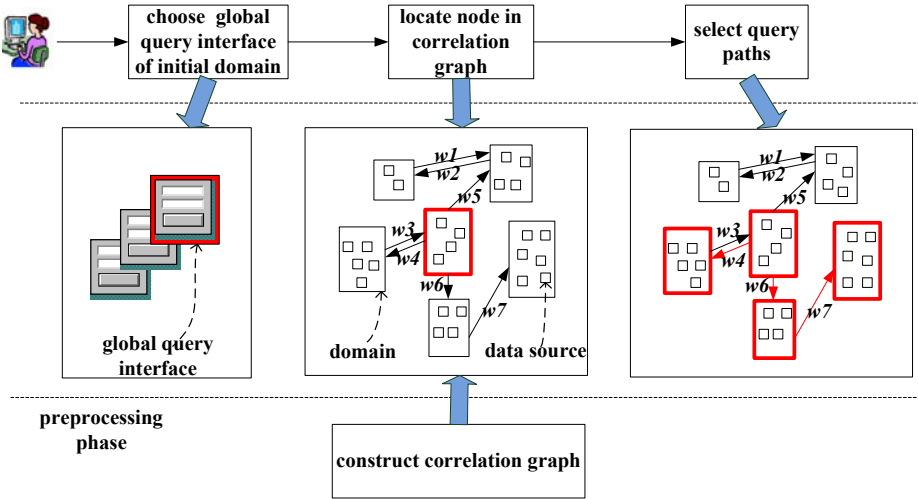
Fig. 1. The framework of multiple-domain linked query path selection

3.2   Entity Class Model

Because of the heterogeneous characteristics between domains and differences between data sources even in the same domain, we use an entity class model to express information of each domain, and correlations between domains are mined by this model. We found that data information provided by each data source can be expressed by an entity class and entities. Take bookshop websites for example, each book represents an entity, so a bookshop website can be expressed as a book entity class.

Given a domain D, it can be expressed in the form of $D_{name}:\{A_0: A_1, A_2,..., A_n\}$, where D_{name} is the domain name, and entity class information includes entity class name A_0 and description attributes $A_1,...,A_n$ which are sorted in descending order by attribute importance. Still take the book domain for example, it can be expressed as Bookshopping:{Book: Book Name, Author, Publisher, Publication Date, ISBN, Price, Abstract}. The entity class model of each domain is defined manually.

4   Solution

4.1   Constructing Correlation Graph

Correlation is a directed weighted graph in which each node represents a domain. From the structure of correlation graph we could see that constructing correlation graph needs two steps: 1) mining correlations 2) computing correlation degree.

(1) Mining Correlations

We use k-nearest neighbor (KNN) classification algorithm to mine correlations between domains and their directions. For each domain pair $<D_i, D_j>$, its training tuples consist of 7 elements and are expressed as ($CAPI$, CAI_i, CAI_j, $CAPS$, $CAPN$, DOD_{ij}/RRR_{ij}, DOD_{ji}/RRR_{ji}), we will introduce these elements in detail.

Definition 1. Attribute Importance. The weight of an attribute is estimated by computing AFDs [8] of a sample dataset.

Definition 2. Attribute semantic similarity. We construct a semantic namespace which consists of semantic hierarchy trees structured by hypernym-hyponym relations. The semantic similarity of an attribute pair is defined by attributes' distance in the semantic hierarchy trees.

Definition 3. Common Attribute. Given two domains $D_1:\{A_0: A_1, A_2,..., A_m \}$ and $D_2:\{B_0: B_1, B_2,..., B_n \}$, we compute the semantic similarity of each attribute pair (A_i, B_j). When a pair of attributes' similarity is over the threshold τ, we call them as a pair of common attributes. Notice that one attribute cannot be a common attribute with multiple attributes simultaneously.

Element 1. Common Attribute Pair Importance (CAPI). It is the importance of common attribute pairs of two domains, and it is defined as Formula 1:

$$CAPI = \sum_{i=1}^{n} \overline{I}_i \tag{1}$$

where n is the number of common pairs, and $\overline{I}_i$ is the mean importance of the ith common attribute pair.

Element 2. Common Attribute Importance (CAI). It reflects the important degree of all the common attributes in one domain. Suppose one domain has n common attributes with another domain, the CAI is defined as shown in Formula 2:

$$CAI = \begin{cases} \left(\sum_{i=1}^{n} I_i , \sum_{i=1}^{n} (I_i - \overline{I})^2 \right) & when\ \forall I_i \neq 0 \\ \\ \left(2 + \sum_{i=1, i \neq j}^{n} I_i, \sum_{i=1}^{n} (I_i - \overline{I})^2 \right) & when\ \exists I_j = 0 \end{cases} \tag{2}$$

where I_i is the importance of the ith common attribute, and $\overline{I}$ is the mean importance of all common attributes. When the entity class name is common attribute, we set its importance as 2 in the summation.

Element 3. Common Attribute Pair Similarity (CAPS). Given a pair of common attributes (A_i, B_j), the CAPS is defined as Formula 7:

$$CAPS = S(A_0, B_j) + S(A_i, B_0) + \sum_{i, j \neq 0} \overline{I}_{i, j} S(A_i, B_j) \tag{3}$$

where $\overline{I}_{i, j}$ is the mean importance of attributes A_i and B_j, $S(A_i, B_j)$ is the semantic similarity of A_i, B_j. When a common attribute pair contains entity class name, the similarity of this pair is directly added to the summation.

Element 4. Common Attribute Pair Number (CAPN). It means the number of common attribute pairs of two domains.

Entity classes of different domains can be divided into two types: public data and user-defined data. Public data refer to entities whose attribute will not be changed by users' operations. Information from different domains may map to one entity, for instance, the movie Titanic and the video tape Titanic essentially refer to the entity of the movie named Titanic. User-defined data refer to entities whose attribute information is defined by users, for example, job, house renting information, and they cannot map to one essential entity. So, we use different ways to reflect the information overlap degree between different domains.

Element 5. Data Overlap Degree (DOD). Given two domain D_1 and D_2 whose data are "public data", S_1 and S_2 are their test sample set respectively, where $|S_1|=m$, $|S_2|=n$. We pick the most important common attribute pair (A_i, B_j) as query condition. For each sample in set S_1, use their values of attribute A_i as query condition value to send queries to domain D_2, and find queries be hit p times, the hit rate p/m is called the data overlap degree(DOD) of domain $D_1 \rightarrow D_2$. Similarly, we could get the data overlap degree of domain $D_2 \rightarrow D_1$.

Element 6. Result Return Rate(RR). Given two domain D1 and D2 whose data are "user-defined data", test sample set and queries are defined the same way as in DOD. For one data source, suppose the default number of results in each page is N, and the number of results in the first page is n, n/N is called full page rate (FP), and result return rate is defined as Formula 4:

$$R = \frac{1}{n}\sum_{i=1}^{n} \overline{FP_i} \tag{4}$$

where $\overline{FP_i}$ is the mean value of data sources' full page rates of the ith query, n is the number of samples in testing sample set.

(2) Computing Correlation Degree

After getting correlations of each domain pair, we need to compute correlation degree. For each correlated domain pair, the correlation degree is defined as Formula 5:

$$Corre = \sum w_i e_i \tag{5}$$

where e_i is the ith element of tuple and w_i is e_i's weight.

How to define the weight of each element is the key point of computing correlation degree. We adopt the combination of Analytical Hierarchy Process (AHP) approach [9] and Rough Sets theory [10] to compute weights, which is the combination of subjective and objective methods.

AHP is a multi-criteria decision making approach in which factors are arranged in a hierarchic structure. We use the method of computing weights of factors for making decisions in the approach. The pairwise comparison matrix for computing weights is given by experts, and weights are computed based on the matrix.

In Rough Sets theory, suppose C is the set of the 5 elements and D is the decision attributes set including correlation and direction. The importance of elements is defined as Formula 6:

$$w_r(e_i) = \frac{r_C(D) - r_{C-ei}(D)}{\sum\limits_{i=1}^{5} r_C(D) - r_{C-ei}(D)} \tag{6}$$

where $r_C(D)$ is the dependency of set D on set C, $r_{C-ei}(D)$ is the dependency of set D on set $C\text{-}e_i$, and the dependency is defined as Formula 7:

$$r_C(D) = \frac{Card(POS_C(D))}{Card(U)} \tag{7}$$

where $POS_C(D)$ is the positive region of set D with set C as its condition attribute set.

Finally, we combine AHP weight and Rough Sets weight together as the weights of elements, as Formula 8 shows:

$$w(e_i) = 0.4*w_a(e_i) + 0.6*w_r(e_i) \tag{8}$$

where $w_a(e_i)$ is AHP weight and $w_r(e_i)$ is Rough Sets weight.

4.2   Selecting Correlated Query Paths

After constructing the correlation graph of domains, we use the idea of Markov Chain to model the process of query path selection. We view the correlation graph as a Markov Chain, with domains as the states, and the weights on correlation edges specifying the probabilities of transition from on state to another. For each node, the weights of all its out edges are normalized corresponding to the definition of transition probability. Using the model of Markov Chain, all the paths correlated to initial domain are picked out. The query path selection algorithm selects all the query paths whose transition probabilities are higher than threshold p on correlation graph G. It uses the depth-first idea to select query paths. Notice that when there are many paths between two nodes, we only keep the longest one.

5   Experimental Evaluation

5.1   Preparation of Data Source Set and Baseline Methods

We used data source set collected by ourselves. First, keywords in thedomain corpus were used as seeds to craw web pages. Then the web pages were filtered to keep those pages containing query interfaces of each domain. Finally, we got 1352 data sources of 67 domains which contain book selling, job hunting, travelling, hotel, restaurant and so on. After extracting entity class model based on query interfaces and result schemas manually, the information-complete data source set was prepared.

Naïve method is an original method of query path selection. The feature words of the initial domain's query interface and results model were extracted, and then computed the overlap degree of its feature words with other domains, if found correlated domains, iteratively searched correlated domains until could not find any. Finally, a query tree was constructed and each branch represented a query path. SCQ method is proposed in [6].

5.2   Performance Evaluation

(1) Evaluation of Correlation Graph Quality

We used the recall and precision of correlation sub-graphs to evaluate the quality of correlation graph. The sub-graphs contained 5, 10, and 15 nodes respectively and were created like this: for the first time, we randomly chose an initial node, and got sub-graphs containing 5, 10, and 15 nodes from the global graph created using our method. Meanwhile, we got the sub-graphs containing the same nodes and got the correlations between them according to the global graph created using SCQ method. For the second time, we got sub-graphs from SCQ global graph first, and then got sub-graphs of ours containing the same nodes as SCQ sub-graphs. Then we manually found all the correlations of these sub-graphs as standard, Figure 2 shows the precisions and the recalls of two methods.
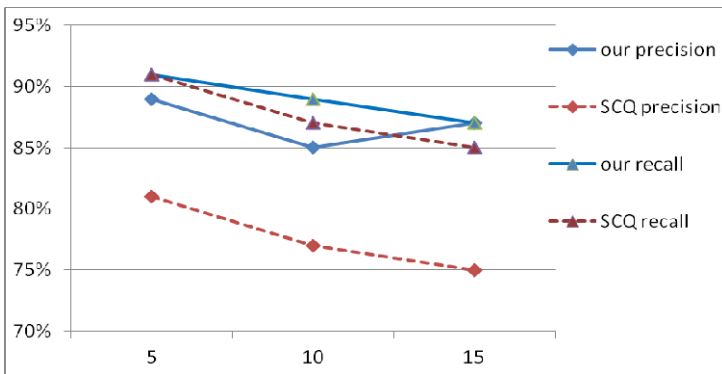
Fig. 2. The precisions and recalls of sub-graphs' correlations
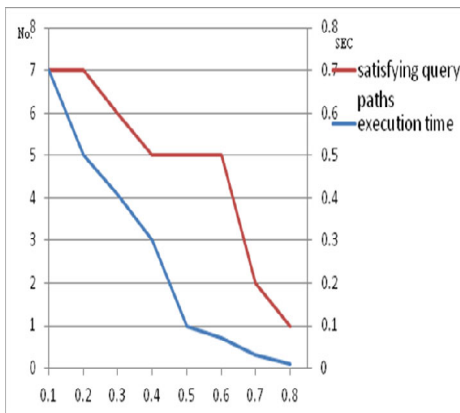
Fig. 3. Variation of execution time and satisfying query paths with threshold p
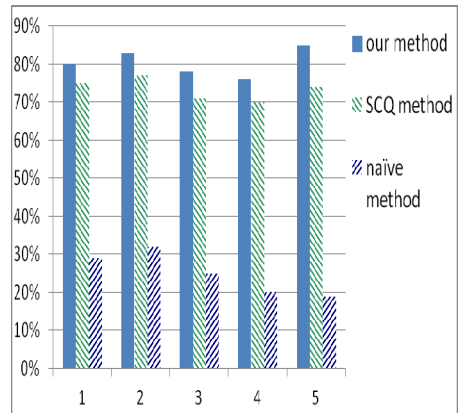
Fig. 4. The correlated query path precisions

(2) Evaluation of Correlated Query Path Quality

First, we introduced how to set threshold p mentioned in Section 4.2. As we can see in Figure 3, as the value of p increased, the execution time and the number of satisfying query paths decreased, so we compromised the quality and efficiency and took the value 0.6.

Next, we compared the precisions of query paths of 3 methods. We randomly chose the initial query domain (5 times in total), and then we judged whether the query paths selected by 3 methods were reasonable. The precisions are shown in Figure 4. We can see that our method and SCQ method are far better than the naïve method, and our method is slightly better than SCQ.

6   Conclusion

We proposed an entity class model based method of correlated query paths selection between multiple domains. Our method is to recommend correlated query paths to users based on initial query domain. We proposed an entity class model to represent domains, and used correlation graph to express the correlations between domains. In order to mine correlations, we defined 6 elements of a domain pair to evaluate 2 domains' correlation. Then we used Markov Chain model to select correlated query paths on the correlation graph. At last we compared our method with others to verify the effectiveness of our method.

References

1. Deutch, D., Greenshpan, O., Milo, T.: Navigating in Complex Mashed-Up Applications. PVLDB 3, 320–329 (2010)
2. Deutch, D., Greenshpan, O., Milo, T.: Navigating through mashed-up applications with compass. In: ICDE 2010, pp. 1117–1120. IEEE Press, New York (2010)
3. Braga, D., Calvanese, D., Campi, A., et al.: NGS: a Framework for Multi-Domain Query Answering. In: ICDE Workshop 2010, pp. 254–261. IEEE Press, New York (2008)
4. Chen, G., Liu, C., Lu, M., et al.: A Cross-Service Travel Engine for Trip Planning. In: SIGMOD 2011, pp. 1251–1253. ACM Press, New York (2011)
5. Bozzon, A., Braga, D., Brambilla, M., et al.: Search Computing: Multi-domain Search on Ranked Data. In: SIGMOD 2011, pp. 1267–1269. ACM Press, New York (2011)
6. Li, Y., Shen, D., Nie, T., Yu, G., Shan, J., Yue, K.: A Self-adaptive Cross-Domain Query Approach on the Deep Web. In: Wang, H., Li, S., Oyama, S., Hu, X., Qian, T. (eds.) WAIM 2011. LNCS, vol. 6897, pp. 43–55. Springer, Heidelberg (2011)
7. Xiao, G., Liu, J.: A Multi-factor Fuzzy Evaluation Model. J. Statistics and Decision 9, 12–14 (2007)
8. Nambiar, U., Kambhampati, S.: Answering imprecise queries over autonomous web database. In: ICDE 2006, pp. 45–54. IEEE Press, New York (2006)
9. Saaty, T.L.: How to Make a Decision: The Analytic Hierarchy Process. European Journal of Operational Research 48, 9–26 (1990)
10. Pawlak, Z.: Rough Sets: Theoretical Aspect of Reasoning about Data. Kluwer Academic Plblishers, Norwell (1992)

ℓ^1-Graph Based Community Detection in Online Social Networks

Liang Huang[1], Ruixuan Li[1,⋆], Yuhua Li[1],
Xiwu Gu[1], Kunmei Wen[1], and Zhiyong Xu[2]

[1] School of Computer Science and Technology, Huazhong University
of Science and Technology, Wuhan 430074, China
[2] Department of Math and Computer Science, Suffolk University, Boston, USA
ellick@smail.hust.edu.cn, {rxli,idcliyuhua,guxiwu,kmwen}@hust.edu.cn,
zxu@mcs.suffolk.edu

Abstract. Detecting community structures in online social network is
a challenging job for traditional algorithms, such as spectral cluster-
ing algorithms, due to the unprecedented large scale of the network. In
this paper, we present an efficient algorithm for community detection
in online social network, which chooses relatively small sample matrix
to alleviate the computational cost. We use ℓ^1-graph to construct the
similarity graph and integrate the graph laplacian with random walk in
directed social network. The experimental results show the effectiveness
of the proposed method.

Keywords: ℓ^1-graph, Spectral clustering algorithm, Graph Laplacian,
Laplacian regularizer.

1 Introduction

Online social networks have attracted more and more attention in recent years.
A typical social network consists of a set of nodes (represents the individual in
the network) and a set of relational tie y_{ij} (measured on each ordered pair of
nodes $i, j = 1, ..., n$). In these social networks, a common feature is the commu-
nity structure. Detecting community structures in social networks is an issue of
considerable practical interest that has received a great deal of attention.

Many algorithms for identifying the communities' structure have been pro-
posed in the past few years. Newman [1] proposed an iterative, divisive method
based on the progressive removal of links with the largest betweenness. Santo
Fortunato [2] developed an algorithm of hierarchical clustering that consists of
finding and removing the edge iteratively with the highest information central-
ity. In addition, Newman and Girvan [3] proposed a quantitative method called
modularity to identify the network communities. It seems to be an effective
method to detect communities in networks. However, Fortunato and Barthélemy
[4] recently pointed out the serious resolution limits of this method and claimed
that the size of a detected module depends on the whole network. To solve this

---

⋆ Corresponding author.

problem, Li [5] developed another quantitative methods called the modularity density. An alternative way to tackle the problem is by spectral clustering [6]. But the algorithm is only effective to handle small networks, when the network's scale gets bigger, it is incompetently for the community detection job.

Compared to the traditional social networks, online social networks have some new features. Firstly, the scale of online social networks are becoming more and more large due to the scale of the internet. The social networks adopted in early community detection algorithms, e.g. the famous benchmark social network Zachary karate club (consists of 34 people in the club), are small social networks. Yet the online social networks such as the LiveJournal consist of tens of millions members and uncountable relationships. Secondly, the relationships between people are becoming more complicated. The instantaneous and random interaction between millions of people have created the uncountable relationship in online social networks. Among these social networks, most are directed one that must be took into consideration.

We integrate the ℓ^1-graph [7] and Laplacian regularizer [8] in our work to handle the situation aforementioned. The work of Chung [9] is also included. This paper introduces these technologies for several good reasons. First, the ℓ^1-graph utilize the overall contextual information instead of only pairwise Euclidean distance as conventionally. The neighboring samples of a datum and the corresponding ingoing edge weights are both considered in the construction of the similarity graph. Second, the laplacian regularizer is suitable for the community detection job in online social network. The laplacian regularizer chooses relatively small sample sets of the given social network. This approach can reduce the computational cost and make it suitable for the online social network. At last, the traditional graph laplacian presume the social network is an undirected and weighted one, makes it unsuitable for the directed social network. As we known, our method is the first work that combine the ℓ^1-graph in the laplacian regularizer when applying in the online social networks. Our studies give a new insight to deal with these tasks and achieves better results compared to the previous outstanding clustering algorithm.

This paper is organized as follows. We introduce our methodology in section 2. Including the details of ℓ^1-graph and the graph laplacian, etc. Subsection 2.3 is the detail of the regularized spectral clustering algorithm. We test the algorithm in the experiments, the results are compared with three algorithm in this section. Finally, the conclusion is given in Section 4.

2   Our Methodology

2.1   Construct Similarity Graph with ℓ^1-graph

There are several popular ways for graph construction like the KNN and the ε-ball, etc. However, these methods only consider the pairwise Euclidean distance without integrating the overall contextual information of the social networks. The ℓ^1-graph uses this information to construct robust and datum-adaptive similarity graph and achieves good results in our experiments.

The construction process of ℓ^1-graph is formally stated as follows.

1) **Inputs:** The sample data set denoted as the matrix $X=[x_1, x_1, ..., x_N]$, where $x_i \in \mathbb{R}^m$.

2) **Robust sparse representation:** The sparse coding can be computed by solving the ℓ^1-Norm optimization problem in (1),

$$\arg \min_{\alpha^i} \|\alpha^i\|, \qquad s.t. \quad x_i = B^i \alpha^i. \tag{1}$$

where matrix $B^i=[x_1, x_1, ..., x_N, I] \in \mathbb{R}^{m+N-1}$.

3) **Graph weight setting:** We can compute the graph weight matrix W for i-th training sample x_i in (2).

$$w_{ij} = \begin{cases} \alpha^i_j, & i > j, \\ 0, & i = j, \\ \alpha^i_{j-1}, & i < j. \end{cases} \tag{2}$$

2.2   Introduction of Undirected and Directed Graph Laplacian

Given the social network $G = (V, E)$, the similarity matrix of the social graph can be denoted with $W = w_{ij}(i, j = 1, ..., n)$, w_{ij} is the similarity between the nodes. Provided G is undirected that $w_{ij} = w_{ji}$, the degree d_i of $v_i \in V$ can be defined as $d_i = \sum_{j=1}^{n} w_{ij}$.

The simplest form of undirected graph laplacian matrix can be defined as

$$L = D - W.$$

D can be defined as the diagonal matrix with the degrees $d_1, ..., d_n$ on the diagonal. (3) is the unnormalized laplacian matrix.

There is another form of graph laplacian matrix called normalized graph laplacians which can be represented as follows

$$L_{sym} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2},$$
$$L_{rw} := D^{-1} L = I - D^{-1} W.$$

A random walk on the given directed graph for graph laplacian is defined in [9]. Given the weighted directed graph G (presume the directed graph G is strongly connected and aperiodic), the out-degree of v_i with d_i^+ can be denoted as follow:

$$d_i^+ = \sum_{j:(v_i, v_j \in E)} w(i, j).$$

The random walk over G with transition probability matrix P can be defined as follows:

$$P_{ij} = \begin{cases} \frac{w(i,j)}{d_i^+}, & if (v_i, v_j) \in E, \\ 0, & otherwise. \end{cases}$$

The above random walk has a unique stationary distribution $\Pi = (\pi_1, ..., \pi_1)^T$ with $\pi_i > 0$ for all i. The Laplacian of G is then defined as

$$\tilde{\mathbf{L}} = \mathbf{I}_n - \frac{\prod^{1/2} \mathbf{P} \prod^{-1/2} + \prod^{1/2} \mathbf{P}^T \prod^{-1/2}}{2}.$$

Where $\prod$ is the diagonal matrix $\prod = diag(\pi_i)$. The Laplacian matrix $\tilde{\mathbf{L}}$ constructed as above can be used in exactly the same way as in the undirected case.

2.3   Regularized Spectral Clustering Algorithm

Several graph regularizers have been proposed in recent years [10,11]. These regularizers usually take the form

$$\mathcal{S}_G(f) = \mathbf{f}^T \mathbf{S} \mathbf{f}.$$

for an appropriate symmetric. The matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ derived from social matrix G. Generally, online social networks may scale up to tens of thousands of nodes and millions of edges. This makes it uncompetitive for standard spectral clustering algorithm due to the frequently matrix operations. The regularized spectral clustering algorithm uses the laplacian regularizer to overcome this shortcoming. By choosing sample nodes in social graph G, the introduced algorithm can alleviate the computational cost significantly through reducing the matrix operation down to acceptable scale.

Instead of using vectors as in spectral clustering algorithm [6], the regularized spectral clustering algorithm constructs a target function for clustering job. This target function then partitions the whole social network naturally. Given have an online social network $G = (V, E), V = v_1, v_2, ..., v_n$, the sample set $G' = (V', E') \subseteq G, V' = v'_1, v'_2, ..., v'_k, k \leq n$ will be chosen to construct the target function, and the function will be used to cluster the whole social network. Let $\mathbf{W}$ be the ℓ^1-weight of G', and L_n be the laplacian matrix of G'. The coefficient α can be calculated by solving the following nonlinear constraint problem (3), see in [12] for details.

$$\alpha = \arg \min_{\alpha \in \mathscr{R}^n} \frac{1}{n^2} \alpha^T \mathbf{W} L_n \mathbf{W} \alpha + \gamma \alpha^T \mathbf{W} \alpha,$$
$$s.t. \quad \frac{1}{n^2} \alpha^T \mathbf{W} D \mathbf{W} \alpha = 1, \tag{3}$$
$$\alpha^T \mathbf{W} D \mathbf{1} = 0.$$

$\alpha^{\mathbf{x}} = (\alpha_1^{\mathbf{x}}, ..., \alpha_n^{\mathbf{x}}) \in \mathscr{R}^n$ is the coefficient of the target function (4). $\mathbf{W}$ is the ℓ^1-graph weight and $\mathbf{1}$ is the vector of all ones. L_n, D is the corresponding laplacian and degree matrix, the details has been introduced in subsection 2.2. One can calculate the target function $\mathbf{f}$ by applying the coefficient in (4).

$$f(x) = \sum_{i=1}^{n} \alpha_i \mathbf{W}(x_i, x). \tag{4}$$

Algorithm 1. The methodology of our paper

Require:
   The social matrix of social network graph $G = (V, E)$;
Ensure:
 1: Random choose the appropriate sample set of the online social network;
 2: Use the ℓ^1-graph to construct the similarity graph of the sample set;
 3: Compute the graph laplacian matrix, if the social network is a directed one, use the method described in [9] to compute the directed graph laplacian matrix;
 4: Compute the generalized eigenvectors and eigenvalues of the graph laplacian matrix;
 5: Use the vectors of the first k eigenvalues to compute the corresponding coefficient α in (9);
 6: Compute the target function values of the whole social network data correspond to the respective coefficient α;
 7: Use the k vectors of the target function values to cluster the social network;
 8: **return** The clusters of the given social network.

Let P be the projection onto the subspace of $\mathbb{R}^u$ orthogonal to $\mathbf{W1}$, one obtains the solution for the constrained problem in (3), which is given by the following generalized eigenvalue problem [8].

$$P(\gamma\mathbf{W} + \mathbf{W}L\mathbf{W})PV = \lambda P\mathbf{W}^2 PV. \tag{5}$$

The final solution is given by $\alpha = PV$, where v is the eigenvector corresponding to the eigenvalues. Similar to the standard spectral clustering algorithm, we choose the eigenvectors of the smallest k eigenvalues to get the k cluster results.

3 Experiments

We test our method in three different social networks, the Arxiv HEP-PH collaboration network [13] and two benchmark social networks [14,15]. The concept of modularity function [1] is employed to quantify the clustering results.

3.1 Two Benchmark Social Networks

In this subsection, four algorithms are tested for the comparison including Kmeans, standard spectral clustering algorithm , regular spectral clustering algorithm with Gaussian similarity graph ($exp(-\|x_i - x_j\|^2/(2\sigma^2))$, $\|x_i - x_j\|$ is the Euclidean distance which be defined in this paper as the shortest distance between node i and j), and our method. These algorithms are applied in two benchmark social networks, the Zachary karate club network [14] with 34 nodes and Dolphin network [15] with 62 nodes. We sample the full networks for regular spectral clustering algorithm in the experiments for comparison.

Figure 1 shows the modularity scores of the four algorithms in Zachary and Dolphin network. Our method outperform the other algorithms in the Zachary
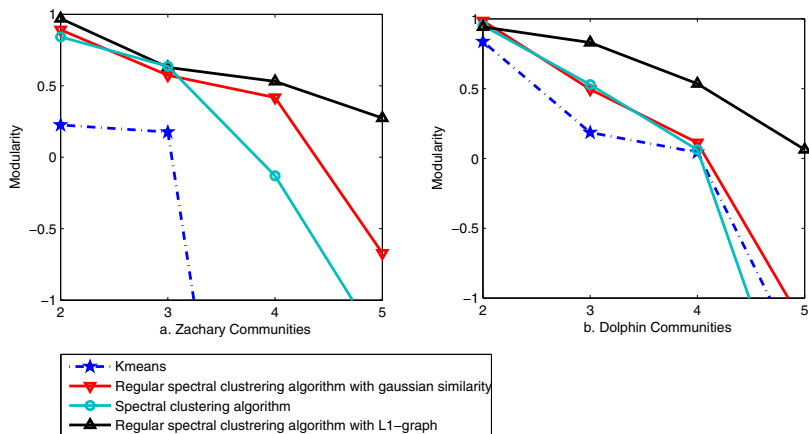
Fig. 1. The modularity of four algorithms applied in Zachary and Dolphin network

network for each community numbers. The standard clustering algorithm achieves the highest score in the dolphin network for but drop fast when the community numbers increase. Our method still win the highest scores for nearly every community numbers in Dolphin network, and gives the most stable performance in Zachary and Dolphin networks both. Kmeans gives the poorest performance in two networks compared to the other algorithms.

3.2   Arxiv HEP-PH Collaboration Network

Arxiv HEP-PH (High Energy Physics - Phenomenology) collaboration network is from the e-print arXiv and covers scientific collaborations between authors papers submitted to High Energy Physics - Phenomenology category. We choose the data covers papers in the period from January 1993 to April 2003 (124 months). A collaboration network with 12,008 nodes and 237,010 edges is constructed in this way.

We choose training sets with size $m \in \{800, 2000\}$ in the experiments, the standard clustering algorithm is employed for the comparison. The most time-consuming part of our method is the computation of the ℓ^1-graph and the shortest distance. In the experiments, it takes about 9 minutes for 800 sample nodes when applying our method from step 2 to 7 in algorithm 1, and 17 minutes for 2000 sample nodes. The standard clustering algorithm takes exceed half an hour for the similar steps. 46 communities are detected in wiki-vote social network corresponding to 500 sample nodes, and 91 communities for 2000 sample nodes. Meanwhile, the standard clustering algorithm detect 77 communities which need to calculate the whole network. We can see from Figure 2, our method can achieves better performance with small sample network compared to the spectral clustering algorithm. When the size of the social network scale up, more time will be spent in standard spectral clustering algorithm with poor performance.
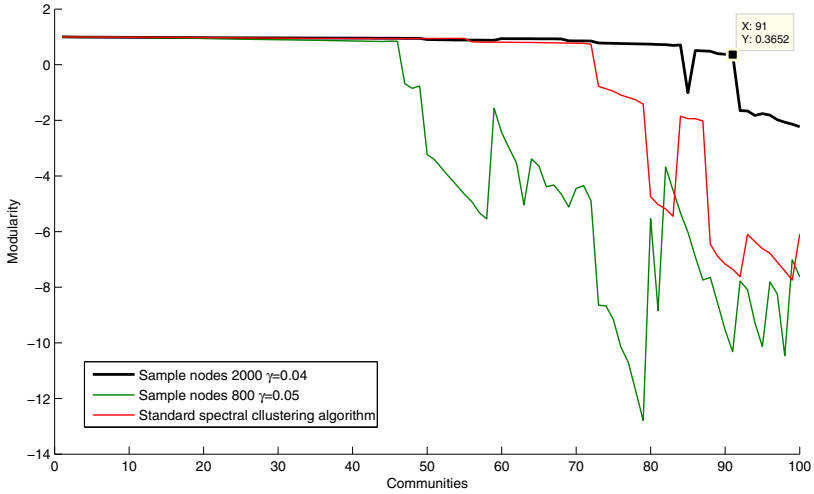
Fig. 2. The three modularity curves of our method (800, 2000 samples) and standard spectral clustering algorithm applied in Arxiv HEP-PH collaboration network

4   Conclusion

The issues of detecting community structures in large online social networks are increasingly common. Our work builds on recent work on ℓ^1-graph and regularization which aimed at detecting community structure in complex networks in an efficient way. The experimental results indicate that our method can achieve better performance with relatively small computational cost.

The method will not only allow for the extension of community structure analysis to some of the very large networks, but will also provide a useful tool for visualizing and understanding the structure of these networks. We hope that this approach will be employed successfully in the search and study of communities in social networks, and will help to uncover new interesting properties in this area.

However, the social networks adopted in our paper are smaller one that consist of tens of thousands nodes. The bottleneck of our method is the computation capability of the computer. We plan to integrate our method into the distributed computing environment such as Map-reduce, etc. We believe that better performance can be achieved in the extreme large online social network like the LiveJounal and delicious social network in this way.

References

1. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E. 69(2), 26113 (2004)
2. Fortunato, S., Latora, V., Marchiori, M.: Method to find community structures based on information centrality. Phys. Rev. E. 70(5), 56104 (2004)
3. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Phys. Rev. E. 69(2), 66133 (2004)
4. Barthélemy, M., Fortunato, S.: Resolution limit in community detection. Proc. Natl. Acad. Sci. 104, 36–41 (2007)
5. Li, Z.P., Chen, L.N., Zhang, S.H., Zhang, X.S., Wang, R.S.: Quantitative function for community detection. Phys. Rev. E. 77, 36109 (2008)
6. Luxburg, U.V.: A Tutorial on Spectral Clustering. Tech. Rep. 149, Max Planck Institute for Biological Cybernetics. (2006)
7. Cheng, B., Huang, T.S., Yang, J.C., Yan, S.C.: Learning With ℓ^1-graph for Image Analysis. IEEE Trans. on Image Processing 19, 858–866 (2010)
8. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. J. Machine Learning Research 7, 2399–2434 (2006)
9. Chung, F.R.K.: Laplacians and the Cheeger inequality for directed graphs. Annals of Combinatorics 9, 1–19 (2005)
10. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. 15(6), 1373–1396 (2002)
11. Smola, A.J., Kondor, R.: Kernels and regularization on grphs. In: Proc. of the 16th Annual Conference on Learning Theory (2003)
12. Cao, Y., Chen, D.R.: Consistency of regularized spectral clustering. Applied and Computational Harmonic Analysis 30, 319–336 (2011)
13. Stanford University Stanford Network Analysis Project, http://snap.stanford.edu/ncp/
14. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. 33, 452–473 (1977)
15. Boisseau, O.J., Dawson, S.M., Haase, P., Lusseau, D., Schneider, K., Slooten, E.: The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. Behav. Ecol. Sociobiol. 54, 396–405 (2003)

Improving Recommendation Based on Features' Co-occurrence Effects in Collaborative Tagging Systems

Hao Han[1], Yi Cai[1,*], Yifeng Shao[1], and Qing Li[2]

[1] School of Software Engineering, South China University of Technology, Guangzhou, China
[2] Department of Computer Science, City University of Hongkong, Hongkong, China
`ycai@scut.edu.cn`

Abstract. Currently, recommender system becomes more and more important and challenging, as users demand higher recommendation quality. Collaborative tagging systems allow users to annotate resources with their own tags which can reflect users' attitude on these resources and some attributes of resources. Based on our observation, we notice that there is co-occurrence effect of features, which may cause the change of user's favor on resources. Current recommendation methods do not take it into consideration. In this paper, we propose an assistant and enhanced method to improve the performance of other methods by combining co-occurrence effect of features in collaborative tagging environment.

1 Introduction

Recommender systems can help overcoming information overload by providing personalized suggestions based on a history of user's behavior [2]. Currently, collaborative tagging systems become more and more popular. Tags provide a simple but powerful way for organizing, retrieving and sharing different types of social resources. Therefore, many recommendation methods have been applied to collaborative tagging systems. Most of these methods in collaborative tagging systems have to analyze users' past behaviors and model user profile. Although these user profiling methods have been widely used in collaborative tagging system, one important problem which is often be ignored is when measuring the favor degree of one user on a resource, the traditional user profiling methods just consider each feature (tag) of resources independently. These methods ignore the following case: one user may like two features when they occur alone, however, when a resource possesses both two features, the favor degree of the user may be changed. The change may be positive or negative.

We consider this change is caused by the *co-occurrence effect* of resources. The effect of two features when they co-occur in a common resource is named by "co-occurrence effect". Co-occurrence effect can be divided into positive and negative. In collaborative tagging systems, rating can be used to indicate the positive co-occurrence effect or negative co-occurrence effect of some features. It will reduces the accuracy of recommendation if we just consider each feature independently without considering their co-occurrence effect.

Therefore, we consider that if we add the co-occurrence effects of features' into recommendation, the performance of recommender system will be improved. Stem from

---

* Corresponding author.

this purpose, we propose an improving recommendation method by combining with feature co-occurrence effects in collaborative tagging systems. The mechanism of our method is as follows. Firstly, we use a *feature co-occurrence effectiveness* to describe the degree of feature co-occurrence effects. Generally speaking, for different user the feature co-occurrence effectiveness may be different. Secondly, we form a user feature co-occurrence effectiveness matrix for each user. Thirdly, we combine our method with another recommendation method to make the final recommendation, and experiment shows that the hybrid method we proposed performs better than using a single existing method only.

The new features of the proposed method and the contributions of our work are as follows. The traditional methods overlook the feature co-occurrence effect, however, which is existing in fact and is related to the accuracy of recommendation. In our work, we make recommendation with considering the effect. Our method is an assistant and enhanced method, hence it can be added to any method. In this way, we combining the advantages of both our method and any other method in together. After combining with our method, the performance of any other method has an obvious improvement.

2   Background and Related Work

In this section, we first review existing work on user profiling methods in collaborative tagging system. Then we discuss the limitations of current works.

2.1   Recommendation Methods

Existing research on recommendation methods can be divided into three main types: *content-based*, *collaborative filtering*, and *hybrid method*.

Content-based (CB) methods provide recommendations by comparing representations of content contained in an item to representations of content that interests the user [8]. On the other hand, collaborative filtering (CF) recommendation methods predict the preferences of active users on items based on the preferences of other similar users or items [4]. The above two methods both have advantages and disadvantages. To avoid the disadvantages of existing approaches, some researchers combine these two methods and introduce the hybrid filtering approach [1].

Compared with CB and hybrid methods, CF methods are more often implemented. Among all the CF methods, Matrix factorization methods [7] become popular recently, some works show that these kinds of methods can obtain a better results than other methods by using Netflix data [3] and Movielens data [6].

2.2   User Profile Modeling in Collaborative Tagging System

Collaborative tagging systems allow users to annotate resources with their own tags, which provide a simple but powerful way for organizing, retrieving and sharing different types of social resources. In collaborative tagging systems, many methods need to model the user according to past user behavior. Vector Space Model (VSM) is frequently used as a representation of user profile. The weight of each tag in VSM is a

degree to which a user is interested in the tag. In VSM, users are mapped to tag vectors in a universal term space. In this vector, the value of weight of each tag is set in [0, 1] , if the value is much closer to 1, it means that the user is more favor on this feature. Generally, the weight of each tag in a user profile (vector) is calculated by term frequency (TF), or term frequency-inverse user frequency (TF-IUF). Another method for user profiling is normalized term frequency (NTF) [5]. For a given tag x, NTF is the possibility or proportion of a user using x. We regard NTF more appropriate to reflect how much the user is interested in x. The resource profiles are similarly constructed based on NTF.

However, using the current user profiling methods may be rather insufficient in collaborative tagging system. There exists one limitation in current user profiling methods without considering co-occurrence effect.

2.3 Limitations of User Profiling Methods without Co-occurrence Effects

As we known, the current methods have only considered the favor features of users independently, and compare it with resource's features, regardless of the feature co-occurrence effects which may makes the user be more interested on this resource or reduces the user's favor and even cause allergy. Let us consider the following example.

Example 1. For a user Bob, if Bob is an active user who frequently buys food online. Through the analysis of his history purchase records, it can be found that he likes "spicy" food and "chicken", but he extremely hates the co-occurrence of "spicy" and "chicken" (dislike spicy chicken), so we give high weights on "spicy" and "chicken" and low weights on the other features. Here the co-occurrence of "spicy" and "chicken" may causes the allergy of Bob, which is a negative co-occurrence effect for Bob, however current methods cannot indicate this information. Then the user profile of Bob can be modeled based on the current user profiling methods as follows:

$$\overrightarrow{U}_{Bob} = (spicy : 0.85, chicken : 0.79, ..., sweet : 0.4)$$

Now there is a Chinese dish named KungPao 'chicken', which is made by 'chicken' and has a 'spicy' taste, so the dish can be modeled as follows:

$$\overrightarrow{T}_{dish} = (spicy : 0.80, chicken : 0.9, ..., sweet : 0.6)$$

Based on these two profiles, this dish will be given a pretty high score and may be recommended to Bob. However, Bob dislikes the resources which possess both 'spicy' and 'chicken', thus this is a bad recommendation for Bob.

Therefore, to the best of our knowledge, current methods are insufficient in processing the co-occurrence effects of features in collaborative tagging systems. This observation gives us an inspiration: If we take the co-occurrence effects into consideration, the recommendation or prediction may be more accurate.

3   User Profiling With Co-occurrence Effect

This section introduces how to represent a user profile by with considering the co-occurrence effects of resource features.

3.1 Modeling User Profiles

The co-occurrence effect mentioned in Section 2 is the co-occurrence of two features. In the collaborative tagging system, a user can give tags to resources as his or her comments and these tags may reflect the preferences of the user. The traditional user profiling methods only represent the favor degree of features independently. However, these methods overlook a detail that when two features, which the user favors when they occur alone, co-occur in a common resource, that may cause the change of the favor degree of the user on this resource, even cause the user's nuisance. In order to handle this problem, we propose a *user feature co-occurrence effectiveness matrix*.

First We can definite the feature co-occurrence effectiveness matrix as follows:

Definition 1. *A feature ooccurrence effectiveness matrix of user i, denoted by M_i, is a matrix of features versus features in which each cell of M_i denoted by $M_i^{f_1,f_2}$ is the co-occurrence effectiveness between two features.*

The value of each cell (co-occurrence effectiveness) is in [-1,+1]. If the value is much closer to +1, it means that the effectiveness is larger, and the co-occurrence of these two features will cause a greater positive (favorite) effect on items to user. If the value is much closer to +1, the result is contrary.

Constructing Feature Co-occurrence Effectiveness Matrix. The user feature co-occurrence effectiveness matrix is used to store all the co-occurrence effectiveness between each two features. Using the matrix we can overcome the limitations of current methods without considering co-occurrence effect. We use an algorithm to introduce the universal method for constructing the matrix. We first calculate the average rating of two features: f_i and f_2 for a particular user i, i.e., the average rating of all resources which are rated by user i meanwhile possess both f_i and f_2. This average rating of f_i and f_2 can be considered as the degree of the co-occurrence effect of f_i and f_2 on user i. The procedure of constructing the matrix is described in Algorithm 1.

In Algorithm 1, if the possible rating score is [0.5,5.0], then Max_{rating} is 5 and Min_{rating} is 0.5. $N_r^{f_1,f_2}$ means the number of times f_1 and f_2 co-occur in the resources rated by score r. Note that if f_1 and f_2 just co-occur few times in some resources which are given a high score by user i, there exist errors in R(f_1, f_2) function, i.e., the number of times of their co-occurrence in the common resources is too small, we don't have enough confidence to assert their co-occurrence cause the high score on these resources. In order to handle this problem, we use *confidence* as a threshold and an impact factor $I_r^{f_1,f_2}$ to judge whether the co-occurrence of f_1 and f_2 will cause a high score or not. Confidence is a default value in [0,1] and in our experiment we set the confidence = 0.3. The impact factor of f_1 and f_2 in the resources which are rated at score r can be calculated as follows:

$$I_r^{f_1,f_2} = \begin{cases} 1 & \frac{2num_r^{f_1,f_2}}{T_r} \geq confidence \\ 0 & x < confidence \end{cases} \tag{1}$$

where T_r is the total number of tags of resources which are rated at score r by user i. Then we make $R(f_1, f_2)$ normalized in [-1,+1] and obtain the co-occurrence effectiveness between two features at line 11.

Algorithm 1. Matrix Generation Algorithm

```
 1: M_i ← ∅;
 2: for each feature denoted by f_1 in the collection of features used by user i; do
 3:     for each feature denoted by f_2 in the collection of features used by user i; do
 4:         if f_1 = f_2 then
 5:             M_i^{f_1,f_2} ← co- effectiveness(f_1, f_2) ← 0;
 6:         else
 7:             R(f_1, f_2) ← (∑_{r=MinRating}^{MaxRating} N_r^{f_1,f_2}·r·I_r^{f_1,f_2}) / (∑_{r=MinRating}^{MaxRating} N_r^{f_1,f_2})
 8:             if R(f_1, f_2) = 0 then
 9:                 M_i^{f_1,f_2} ← co- effectiveness(f_1, f_2) ← 0;
10:             else
11:                 M_i^{f_1,f_2} ← co- effectiveness(f_1, f_2) ← (2(R(f_1,f_2)−Mid_{rating})) / (Max_{rating}−Min_{rating});
12:             end if
13:         end if
14:     end for
15: end for
```

4   Recommendation Process With Co-occurrence Effect

In collaborative tagging systems, not every user will give more than one tags (features) on a common resource, which results in the sparsity of the user feature co-occurrence effectiveness matrix. Further, there are some resources cannot be predicted by our method. But for each resource which can be predicted by our method, the prediction is accurate. When combined with any other recommendation methods, our method will obtain a better result than using these methods only. Our recommendation method is represented as follows:

$$RScore(i, j) = \beta(\alpha_1(i, j), \alpha_2(i, j))$$

where *RScore(i,j)* is the predicted rating for user i to resource j, $\alpha_1(i, j)$ is the predicted rating for user i to resource j by our method, and $\alpha_2(i, j)$ is the predicted rating for user i to resource j by other method. Next we will introduce how to represent $\alpha_1(i, j)$ and the algorithm to measure RScore.

4.1   Feature Co-occurrence Effectiveness Function $\alpha_1(i, j)$

To make all features' co-occurrence effectiveness into consideration, we make an aggregation on all cell value of the feature co-occurrence effectiveness matrix into $\alpha_1(i, j)$. There are many methods to construct the feature co-occurrence effectiveness function, but here we use a method that finding the summation(i.e., $\frac{\sum M_{f_1,f_2}^i}{2}$). M_{f_1,f_2}^i is the co-occurrence effectiveness between f_1 and f_2. Then we map α_1 to [0,5.0] in accordance to the rating scale as follows:

$$\alpha_1(i, j) = 5 * (1 - \frac{2\arctan(\frac{\sum M_{f_1,f_2}^i}{2}) + \pi}{2\pi})$$

where $\alpha_1(i, j)$ is the predicted rating from the user i to resource j.

4.2   RScore Measure Algorithm

There are many methods to construct *RScore* function by the use of $\alpha_1(i,j)$ and $\alpha_2(i,j)$. Here we use a brief words to introduce the RScore measure algorithm. If a resource can be predicted by our method, then the predict score comes from our method. Otherwise, it comes from other method.

5   Experiments

The proposed method can be used as an enhanced counterpart of existing methods. Thus, we conduct experiments by combining our method with two recommendation methods which are *content-base (CB)*, and *matrix factorization (MF)* respectively, to evaluate to what degree our method can improve the performance of these methods.

5.1   Experiment Setting

1)Data Set Description: To evaluate our recommendation method, we use the Movie-Lens data set in our experiments, and this data set is widely used in previous papers. MovieLens data set used in our experiments contains 44805 user-movie-tag-rating tuples, annotated by 2025 users on 4796 movies. Each user has rated various number of movies, and the rating follow the 0.5 (bad) - 5 (excellent) numerical scale. Each tag is typically a single word, or a short phrase. The meaning and the purpose of particular tag are determined by each user. We use 80 percent of user-movie-tag-rating tuples as the training set and the rest 20 percent tuples as the test set.

2)Metrics: To evaluate the efficiency of our method, two metrics are employed here for evaluation. The first one is *mean absolute error* (MAE) defined as the average absolute difference between predicted ratings and actual ratings. For the MAE value of a recommendation method, the smaller the better. The second one is *Root Mean Squared Error*(RMSE). It is a frequently used measure of the difference between predicted ratings and the actual ratings.

3)Baseline Methods: We combine our method with CB and MF respectively.[1] We dub these two hybrid methods "CB++" and "MF++". To evaluate the improvement of our method on any other methods, we make two comparisons. The first comparison is between CB and CB++, the second comparison is between MF and MF++.

5.2   Experiment Result

5.2.1   Comparison between CB and CB++

In order to find the optimal weight of our method in CB++, we evaluate the effect of values of parameter θ by setting 11 different values (0, 0.1, 0.2, ..., 1.0) for θ in Algorithm 2.

---

[1] Matrix factorization is the one of state-of-the-art methods, which is inspired by CF, which outperforms other CF methods in recommendation systems. Therefore, we select MF as a baseline method. [7].
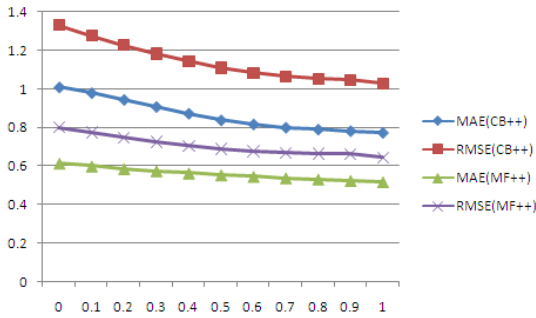
Fig. 1. MAE and RMSE with different weight of our method for CB++ and MF++
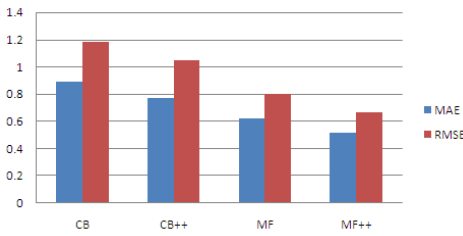
Fig. 2. Comparison between CB, CB++, MF, and MF++
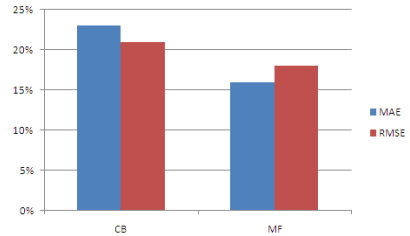
Fig. 3. Improvements of CB++ on CB and MF++ on MF

From Figure 1, we can find that if the weight of our method equals to 1.0, the MAE and RMSE are smallest. Figure 2 shows the comparison between CB and CB++ (θ in our method is set as 1.0). The MAE and RMSE for CB are 0.89 and 1.18, respectively. For CB++, the MAE is 0.77 and the RMSE is 1.03. The improvement of CB if it is combined with our method can be seen from Figure 3. The improvement of CB++ comparing with CB on MAE is 23% and on RMSE is 21%.

5.2.2 Comparison between MF and MF++

Similarly, we set 11 different weights (weight = 0, 0.1, ...,1.0) for θ to conduct experiments on MF++. The result is shown in Figure 1. When the weight of θ in our method is 1.0, we can obtain a best result. According to Figure 2, the MAE and RMSE are 0.62 and 0.80 by using MF, which are desired results. Nevertheless, there is a huge margin of improvement if we make our method as a supporting facilities for MF. Using MF++ (with $\theta = 1.0$), we can obtain the values of MAE and RMSE are 0.52 and 0.67, respectively. From Figure 3, we can find that the improvement of MF++ comparing with MF on MAE is 16% and on RMSE is 18%. Based on the above two comparisons, we can conclude that our method can combine with existing methods well and enhance existing methods to obtain a better recommendation result.

6   Conclusion

In this paper, we focus on exploring how to improve the recommendation with features' co-occurrence effects. We model the profile of a user from a new perspective, by constructing a feature co-occurrence matrix of the user in collaborative tagging environment. There are two main differences between our method and previous methods. One is that when we model a user, we take the effects of features' co-occurrence into consideration. The other difference is that our method is an assistant and enhanced method, and it can be used to improve existing methods in collaborative tagging systems. To the best of our knowledge, our work is the first effort on this direction.

References

1. A Survey of E-Commerce Recommender Systems (June 2007)
2. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. 17(6), 734–749 (2005)
3. Bennett, J., Lanning, S., Netflix, N.: The netflix prize. In: KDD Cup and Workshop in conjunction with KDD (2007)
4. Cai, Y., Leung, H.F., Li, Q., Tang, J., Li, J.: Tyco: Towards typicality-based collaborative filtering recommendation. In: ICTAI (2), pp. 97–104. IEEE Computer Society (2010)
5. Cai, Y., Li, Q., Xie, H., Yu, L.: Personalized Resource Search by Tag-Based User Profile and Resource Profile. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 510–523. Springer, Heidelberg (2010)
6. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd ACM SIGIR Conference, August 15-19, pp. 230–237 (1999)
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42, 30–37 (2009)
8. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence, pp. 187–192 (2002)

Memory Performance Prediction of Web Server Applications Based on Grey System Theory

Faliang Huang[1], Shichao Zhang[2], Changan Yuan[3], and Zhi Zhong[3]

[1] Faculty of Software, Fujian Normal University, Fuzhou 350007, P.R. China
faliang.huang@gmail.com
[2] Faculty of Engineering and Information Technology, University of Technology,
Sydney, P.O. Box 123, Broadway NSW 2007, Australia
[3] Science Computing and Intelligent Information Processing
of GuangXi Higher Education Key Laboratory, Nanning 530023, China

Abstract. With the success of internet, recently more and more companies start to run web-based business. While running e-business sites, many companies have encountered unexpected degeneration of their web server applications performance, which may lead to loss of customers. Many managers wish to have a decision-support tool that cancan answer such questions, such as "will my web server applications performance degenerate?", and "what are the main reasons of the degenerations?". In this paper we first propose a new memory performance prediction model of web server applications based on grey system theory. And then, a software system "Memory Performance Manager" (MPM) is developed for predicting memory performance of the web server applications. Massive experiments demonstrate that the effectiveness of MPM's in predicting web server memory performances.

Keywords: Grey system, web server applications management, memory performance prediction.

1 Introduction

With the success of internet, more and more companies have started to run web-based business. However, not all companies have succeeded in achieving their desired goals. An e-business site without great user experience would have less people to visit, as eventually will make it difficult for the company to survive in market competition. So many companies desire to acquire a tool that can effectively monitor and predict performance of their web applications. That is to say, they wish to have some expert system that can answer such questions "will web server applications performance degenerate?", "what are main reasons of the degenerations?", etc. Performance of web server applications depends on various factors, and among them the web server memory management is a major one. Therefore efficient management of web server memory is very important to companies that run web-based business[1,2,3].

Intuitively, web server applications performance degradation should be detected by memory performance testing. However, the complexity of modern web server applications has made the number of bugs and defects in web server applications

incredibly increase, and hard to identify and resolve. So it is impractical to remove all kinds of performance degradations of web server applications by software testing.

With an attempt to address the issues, we proposed a novel web server memory performance prediction model based on grey system theory [4], and developed a software system MPM. The MPM can continuously collect stream data from system monitor counters in the operating system running web server applications and dynamically analyze the collected data to discover hidden trends of web server memory performance.

It is important to mention that although MPM has something in common with usual software testing tools, there is a remarkable difference between them. The latter is mainly used in finding bugs prior to software release, but the former is used to predict performance and identify potentials of web server applications in service. Our experiments have demonstrated MPM's effectiveness in predicting memory performance of web applications and MPM can provide great help for e-business website managers to efficiently manage web server applications.

2 Related Work

Our work is closely related to grey system theory[4], which is initiated by Deng is mainly utilized to study uncertainties in system models and make forecasts and decisions. The representative prediction model GM(1,1) has been successfully used to model dynamic systems in different fields such as industry, economy, software engineering, environment protection, and so on[5,6,7,8]. It can make prediction in poor, incomplete or uncertain circumstances in a system without the need of long-term historical data.

Another related work is web applications performance testing, which is an important tool for web application developers to predict performance characteristics of an application in production. Recently, web server applications performance management has increasingly drawn attention of researchers. Mosberger and Jin proposed a tool httperf[9] to measure web server performance. Schroeder and Harchol-Balter[10] presented a SRPT scheduling policy to reduce response time of a web server. A transductive inference based algorithm was designed to predict system performance based on limited historical testing data[11]. Although the aforementioned performance testing approaches effectively reduce the risks of web server applications performance degenerations, the complexity and variability of environment running web server applications make it impossible to completely eradicate performance degenerations of web server applications with performance testing. Therefore existing performance testing tools cannot satisfy the needs of e-business companies.

3 Memory Performance Manager

3.1 Framework of MPM

The framework of MPM (fig.1) primarily consists of five components: System Performance Monitor, Data Preprocessor, Counter Value Predictor, Memory

Performance Analyzer and User Interface. In this section we will briefly describe the workflows of MPM.

To use MPM, a user should input parameters such as time granularity parameter in the User Interface. The System Performance Monitor collects real-time system performance data of the target web server. Thereafter the collected system performance data is transferred to Data Preprocessor, which would fulfills data pre-treatment in accordance with user demands. Then Counter Value Predictor would use the preprocessed data as historical knowledge and predict server memory performance in user-defined way. Finally, Memory Performance Analyzer would evaluates the future performance of sever memory and produce the final test reports or advices to the user.
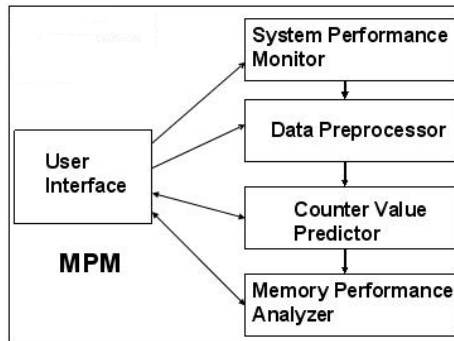
Fig. 1. Framework of MPM

3.2   User Interface

The User Interface component is responsible to facilitate communication between the user and the system MPM. With User Interface component, users can select some special system performance monitors for system performance data acquisition and assign parameters, such as prediction granularity and prediction time interval, for related algorithms.

3.3   System Performance Monitor

The function of module System Performance Monitor is to track and record system performance. Users can choose target counters related to system memory performance to construct a counter container in the user interface module. In this paper, we choose the following 10 counters: Available Mbytes(AM), Pages/sec(PS), Page Reads/sec(PRS), Page Faults/sec(PFS) , Page Input/sec(PIS) , % Disk Time(DT) , Avg. Disk Queue Length(ADQL), Private Bytes(PB), Working Set(WS), Handle Counter(HC) .

3.4   Data Preprocessor

The complexity of web server applications running environments may result that the counter value sequence collected by System Performance Monitor is unclean or dirty. In other word, some random factors such as occasional competition of running

processes in a system may cause sharp and instantaneous but not enduring increase in some counters values. Since what we pay attention to is general trend, we treat the data above 95th percentile and below 5th percentile as outliers and remove them. After removing the outliers, we further conduct data generalization according to the time granularity parameter g given by the user. There are lots of techniques available to generalize data, and here we adopt the simple "mean" strategy. Let the initial counter value sequence $X = \{x(1), x(2), \cdots, x(i), \cdots, x(n)\}$, resultant sequence X' can be computed according to formula 1 and 2.

$$K = \left\lceil \frac{n}{g} \right\rceil \tag{1}$$

$$X' = \begin{cases} \{\dfrac{1}{g}\sum_{i=1}^{g}x(i), \cdots, \dfrac{1}{g}\sum_{i=k*g+1}^{(k+1)*g}x(i), \cdots, \dfrac{1}{g}\sum_{i=(K-1)*g+1}^{K*g}x(i)\}, \quad k=0,1,\cdots,K \;, if\; n \bmod g = 0 \\ \{\dfrac{1}{g}\sum_{i=1}^{g}x(i), \cdots, \dfrac{1}{g}\sum_{i=k*g+1}^{(k+1)*g}x(i), \cdots, \dfrac{1}{g}\sum_{i=(K-1)*g+1}^{K*g}x(i), \dfrac{1}{g}\sum_{i=K*g+1}^{n}x(i)\}, k=0,1,\cdots,K, otherwise \end{cases} \tag{2}$$

3.5   Predicting Counter Value with Grey Model EGM(1,1)

In MPM, we use EGM(1,1) [12] to predict counter value based on the following considerations: Web applications running environments are very complex, changing and unpredictable, which make it nearly impossible to collect equidistance counter value sequence in such environments, and further make traditional GM(1,1) model unfit for the prediction. In contrast with GM(1,1), main improvement of EGM(1,1) is the background value equation as below

$$Z^{(1)}(k+1) = \frac{1}{2m}((m+1)x^{(1)}(k) + (m-1)x^{(1)}(k+1)), \; k=1,2,\cdots,n-1 \tag{3}$$

$$m = \left(\sum_{i=2}^{n} \frac{x^{(1)}(i)}{x^{(1)}(i-1)} \right)^{\frac{1}{n-1}} \tag{4}$$

3.6   Memory Performance Analyzer

Obviously, it is nearly impossible for an e-business web site manager to make a judgment about memory performance of the running web server applications with EGM. So we present the module memory performance analyzer with MPReasoner as core to better support the managers' decisions. This analyzer is to check if memory performance degeneration will occur within a certain time range given by an e-business web site manager according to the following heuristic rule: if predicted value of any counter violates its responding threshold, this means that there exists memory performance degeneration in the target e-business web site. Note that although there are many reasons of performance degeneration, we just consider two factors for simplicity: hard disk defects and memory leak.

Algorithm MPReasoner

If (val(AM)<=TH_AM or val(PS)>=TH_PS or val(PRS)>=TH_PRS or val(PIS)/ val(PFS) >=TH_ PISiPFS) and (val(PRS)<TH_PRS) and (val(DT)>=TH_DT) or (val(ADQL)>=TH_ADQL))) reasonList.add("harddisk defects");

For each counter in counter set ={PB, WS, HC}

$$\Delta t = \sum_{i=2}^{n} (x(i) - x(i-1)), \quad i = 2,3,\cdots,n$$

If $\dfrac{\Delta t}{x(1)} \geq 0.1$ reasonList.add("memory leak");

4   Experiments

4.1   Experimental Environments

To testify the effectiveness of our approach, we use LoadRunner to simulate running environments of web server applications. LoadRunner is used to configure test environment, record script and execute the designed tests. The number of events evoked by users is enormous in real e-business website, and we have picked up two most frequent transactions in our experiments: visit of some given page and customization and acknowledgement of orders. For simplicity, we only consider memory performance management of web application running in two different testing environments: changeable load testing and endurance testing. We have designed 12 different experimental scenarios for the two testing environments, which are listed in Table 1 and Table 2. In table 1, PIncRate denotes the rate of user increase (users/minute)

Table 1. Experimental scenarios for changeable load testing

Scenario	#ExpectedUser	PIncRate
S1	1000	10
S2	1000	20
S3	1000	30
S4	1500	10
S5	1500	20
S6	1500	30
S7	2000	10
S8	2000	20
S9	2000	30

Table 2. Experimental scenarios for endurance testing

Scenario	# CurrentUser
S10	1000
S11	1500
S12	2000

4.2   Experimental Results

In order to validate effectiveness of MPM, we introduce a two-phase evaluation: the first phase aims to test prediction precision of component Counter Value Predictor in MPM, and the second phase is to verify the consistency between decisions made by MPM and decisions made by domain experts.

4.2.1   Prediction Precision Evaluation

We have used the following formula to evaluate prediction results:

$$precision = 1 - \frac{\left|x(i) - \hat{x}(i)\right|}{x(i)} \tag{5}$$

Where x(i) is real value of a counter, $\hat{x}(i)$ is the predicted value.

For each scenario from S1 to S9, we have further generated six experiment datasets according to LoadRatio (LoadRatio =#CrrentUsers/#ExpectedUsers). For each dataset, we have done 10 groups of experiments and recorded the average precision of these groups(Fig.2(a)). From Fig.2(a), we can see that in nearly all scenarios (excluding S3), precision increases more or less with LoadRatio increasing, which can be easily explained by the information whitening process: the larger LoadRatio, that is to say, the more similar is current memory performance to memory performance to be predicted, since memory performance becomes much whiter, so the precision becomes much higher.
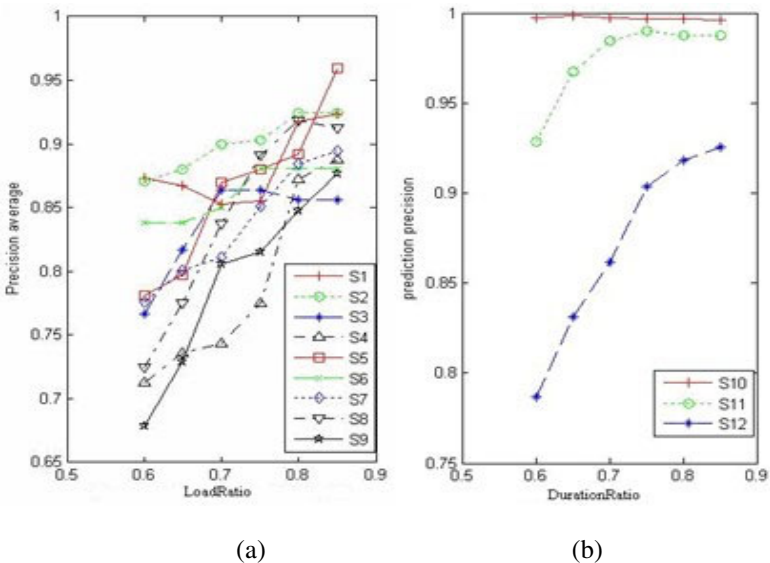
Fig. 2. Prediction results in scenarios: (a)S1 to S9, (b) S10 to S12

Similarly, we have Fig.2 (b) for the scenarios from S10 to S12. From the figure, we can draw a conclusion analogous to the conclusion from Fig.2 (a): Generally, increase of DurationRatio(formula 6) will lead to higher prediction rate, as can be explained by information "whitening" principle. Finally, comparing Fig.2(a) with Fig.2(b) shows that Counter Value Predictor performs much better in the scenarios from S1 to S9 than in the scenarios from S10 to S12, and we think it is because web server running environment in scenarios S1 to S9 is more complex than that in scenarios S10 to S12.

$$DurationRatio = \frac{CSize}{FSize} \tag{6}$$

Where CSize denotes the length of time interval for input sample, FSize denotes the length of time interval from sample starting point to user-defined time point.

4.2.2   Decision Consistency Verification

The final merit of MPM is to provide useful knowledge for web server applications performance management. We hope to know how much MPM can help the managers through decision consistency verification, which is accomplished by comparing MPM's decisions with many experts' decisions. We chose scenario S1 and S10 as verification scenarios and did the following two consistency verification: one is Performance Degeneration Existence Decision Consistency (PDEDC), the other is Performance Degeneration Reason Decision Consistency (PDRDC). Our verification approach can be depicted as follows: first, we get the MPM decisions in scenario S1 and S10, and then ask 15 different experts, including web server manager, web application tester and so on, to independently analyze the data from scenario S1 and S10 with LoadRunner and give their decisions respectively. To quantify the consistency, we construct the following two formulas:

$$PDEDC = \frac{|\,experts\ having\ same\ existence\ decision\ with\ MPM\,|}{|\,experts\,|} \tag{7}$$

Where decision means deciding whether there is memory performance degeneration.

$$PDRDC = \frac{|\,experts\ having\ same\ reason\ decision\ with\ MPM\,|}{|\,experts\,|} \tag{8}$$

Where decision means deciding what is the main cause of the degeneration.

Both table 3 and table 4 show that PDEDC is higher than the responding PDRDC, this is not difficult to understand, because predicting if there is memory performance degeneration is much easier than predicting what is the main cause of the degeneration. After comparing table 3 and table 4 we can reach a conclusion that almost all PDEDCs and PDEDCs in scenario S1 are higher than counterparts in scenario S10, owing to the fact that environments in scenario S1 are more complicated than scenario S10.

Table 3. Decision consistency verification in scenario 1

LoadRatio	PDEDC	PDRDC
0.60	0.93	0.8
0.65	0.87	0.8
0.70	0.8	0.73
0.75	0.8	0.73
0.80	0.73	0.73
0.85	0.73	0.67
Average	0.81	0.74

Table 4. Decision consistency verification in scenario 10

DurationRatio	PDEDC	PDRDC
0.60	0.93	0.73
0.65	0.8	0.87
0.70	0.93	0.87
0.75	0.87	0.73
0.80	0.8	0.73
0.85	0.73	0.67
Average	0.84	0.77

4.2.3   Summary

The above two phase evaluation shows that (i) in the first phase, prediction precision plays an important role in both PDEDCs and PDRDEs, (ii) in the second phase, both PDEDCs and PDRDEs weakly depend on prediction precision. From the above two-phase evaluation, we see that MPM can effectively predict memory performance of web server applications, and provide useful knowledge for web server mangers' decision making. However, in some scenarios, the results are not accurate; we think there are many reasons for the inaccuracy such as counter selection strategy, counter value threshold determination, prediction precision of the first phase and so on.

5   Conclusion

Performance management of web server applications has become increasingly important in both industry and academia. In order to tackle the problem, we proposed a novel web server memory performance prediction model based on grey theory and constructed a useful software system MPM. Through substantial experiments we can draw the conclusion that MPM can effectively predict memory performance for web applications and provide great help in the management of web server applications performance. Howerver, there are still some limitations in our current MPM, for example, under complex scenarios, its reliability becomes low, and many counter value thresholds need to be given manually. How to overcome these limitations and improve MPM's robustness is our future work.

References

1. Liu, Y., Zhu, L., Gorton, I.: Performance assessment for e-government services: an experience report. In: CBSE 2007, pp. 74–89 (2007)
2. Hao, W., Fu, J., Yen, I.-L., et al.: Achieving high performance web applications by service and database replications at edge servers. In: IPCCC 2009, pp. 153–160 (2009)
3. Magalhães, J.P., Silva, L.M.: Root-cause analysis of performance anomalies in web-based applications. In: SAC 2011, pp. 209–216 (2011)
4. Deng, J.: Introduction to grey system theory. J. Grey Syst. (1), 1–24 (1989)
5. Huang, K., Jane, C.-J.: A hybrid model for stock market forecasting and portfolio selection based on ARX, grey system and RS theories. Expert. Syst. Appl. 36(3), 5387–5392 (2009)
6. Zhu, S., Wang, J., Zhao, W.: A seasonal hybrid procedure for electricity demand forecasting in China. Applied Energy 88(11), 3807–3815 (2011)
7. Song, Q., Martin, S.: Predicting software project effort: A grey relational analysis based method. Expert Syst. Appl. 38(6), 7302–7316 (2011)
8. Zhao, J., Wang, W., Liu, Y.: A Two-Stage Online Prediction Method for a Blast Furnace Gas System and Its Application. IEEE T. Contr. Syst. T. 19(3), 507–520 (2011)
9. Mosberger, D., Jin, T.: httperf—A Tool for Measuring Web Server Performance. In: Proceedings of the First Workshop on Internet Server Performance, Madison, WI (1998)
10. Schroeder, B., Harchol-Balter, M.: Web servers under overload: How scheduling can help. ACM Trans. on Internet Technology 6(1), 20–52 (2006)
11. Ma, L., Luo, T., Song, J., et al.: Web performance testing and prediction. Journal of the Graduate School of the Chinese Academy of Sciences 22(4), 472–479 (2005)
12. Tan, G.J.: The structure method and application of background value in grey system GM(1,1) Model (I). J. Theor. Pract. Syst. Eng. 20(4), 98–103 (2000)

Performance Optimization of Analysis Rules in Real-Time Active Data Warehouses[*]

Ziyu Lin[1], Dongzhan Zhang[1,**], Chen Lin[1], Yongxuan Lai[2], and Quan Zou[1]

[1] School of Information Science and Technology, Xiamen University, Xiamen, China
{ziyulin,zdz,chenlin,zouquan}@xmu.edu.cn
[2] School of Software, Xiamen University, Xiamen, China
laiyx@xmu.edu.cn

Abstract. Analysis rule is an important component of a real-time active data warehouse. Performance optimization of analysis rules may greatly improve the system response time when a new event occurs. In this paper, we carry out the optimization work through the following three ways: (1)initiating non-real-time analysis rules as less as possible during rush hour of real-time analysis rules; (2) executing non-real-time analysis rules using the same cube at the same time interval; and (3) preparing frequent cubes for the use of real-time analysis rules ahead of time. We design the LADE system to get all the reference information required by optimization work. A new algorithm, called ARPO, is proposed to carry out the optimization work. Empirical studies show that our methods can effectively improve the performance of analysis rules.

Keywords: analysis rules, real-time active data warehouses.

1 Introduction

In the past decades, data warehouses have been going through five different stages, i.e. reporting, analyzing, predicting, operationalizing and active warehousing. Now real-time active data warehouses [1,2,3,4] are attracting more and more attention due to the great benefits they bring to the organization.

Analysis rule[1] is a very important part of a real-time active data warehouse. It detects the occurrence of events and initiates analysis process, during which multi-dimensional data will be used. If certain condition evaluates to be TRUE, the corresponding action will be triggered, such as sending alerts to analysis workers. Up to date, most of the research work on analysis rule is focused on its mechanism (e.g. [5,1]). In fact, performance optimization of analysis rules is also a critical aspect, though, to the best of our knowledge, there is still no published work on it. If more attention is paid to the optimization work, we can on one

hand make full use of system resources, and on the other hand, achieve better performance for analysis rules.

In this paper, we propose the issue of performance optimization of analysis rules in real-time active data warehouses. Here analysis rules are divided into two types, namely, *real-time analysis rules* and *non-real-time analysis rules*. We define rush hour and frequent cubes for real-time analysis rules, and cube using pattern for non-real-time analysis rules. Our optimization work is focused on three aspects: (1)initiating non-real-time analysis rules as less as possible during rush hour of real-time analysis rules; (2) executing non-real-time analysis rules using the same cube at the same time interval; and (3) preparing frequent cubes for the use of real-time analysis rules ahead of time. The LADE(Log data mining-based Active Decision Engine) system is designed to help get all the reference information required by optimization work, such as rush hour, cube using pattern matrix and frequent cube matrix. Then we give a new algorithm, called ARPO (Analysis Rule Performance Optimization), to carry out the optimization work. We also conduct experiments in LADE system, and the results show that our method can effectively improve the performance of analysis rules in real-time active data ware houses.

The remainder of this paper is organized as follows: Sect. 2 gives problem statement. In Sect. 3, we introduce the LADE system first, followed by the detailed description of getting reference information for optimization work. In Sect. 4, we will show how to carry out the optimization work. The experiment results are reported in Sect. 5. Sect. 6 discusses the related work. Finally, we give the discussion and conclusion in Sect. 7.

2   Problem Statement

Compared to the traditional data warehouse, a real-time active data warehouse usually has an additional component called "real-time data cache", which stores all the real-time data. Through CDC(Change Data Capture), data change occurring in the OLTP system can be captured and propagated to the real-time data cache and active decision engine. The active engine is composed of event model, rule model, action model and meta-data model. Event model detects the occurrence of events, and rule model runs analysis rules after the occurrence of certain event. If a rule evaluates to be TRUE, the corresponding action will be triggered, such as notifying the analysis workers. Such process is called *active decision making*, during which the active decision engine may access the OLAP server for the required cubes. User may define the analysis rules for the active decision engine, and deal with the problems and conflicts that need to be treated manually.

Analysis rules can be classified into two types: real-time analysis rules and non-real-time analysis rules. The former, denoted $\mathcal{R}$, after being triggered, makes real-time decision, which will be fed back to operational system to satisfy real-time business requirements. The latter, denoted $\mathcal{N}$, after being initiated, makes decision usually not for the purpose of real-time applications.

Analysis rules use cubes to carry out multi-dimensional analysis. With the help of *cube using pattern* (see Definition 1), we can initiate, during the same period as much as possible, those non-real-time analysis rules that access the same cube. The detailed information about how a cube is defined can be found in [1].

Also, by generating frequent cubes those are most accessed in certain period ahead of time for a real-time analysis rule $\mathcal{R}$, we can greatly improve the performance of $\mathcal{R}$ in some cases. As far as non-real-time analysis rules are concerned, the concept of frequent cube is not so useful, since they do not make real-time decision.

Problem Statement. Given a set of real-time analysis rules $S_1 = \{\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_m\}$ and a set of non-real-time analysis rules $S_2 = \{\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_n\}$, performance optimization of analysis rules works as follows:

- find the rush hour $\bigcup[t_a, t_b)$, and during these time intervals, initiate the rules in S_2 as less as possible;
- find those rules in S_2 that use the same cubes in the multi-dimensional analysis process, and initiate them at the same period as much as possible;
- find the frequent cubes for certain period $[t_1, t_2)$, and prepare these cubes for the rules in S_1 before t_1.

3   Getting the Information for Performance Optimization

3.1   The LADE System

The LADE system designed by us is used to perform data mining based on the log of analysis rules. As is shown in Fig.1, in LADE, we extend the traditional architecture of active decision engine [5] by adding the logging component, called *action log*, to record all the necessary information about analysis rules, such as *ID*, *IsRealTime*, *RuleInfoID*, *CubeID* and *Time*.
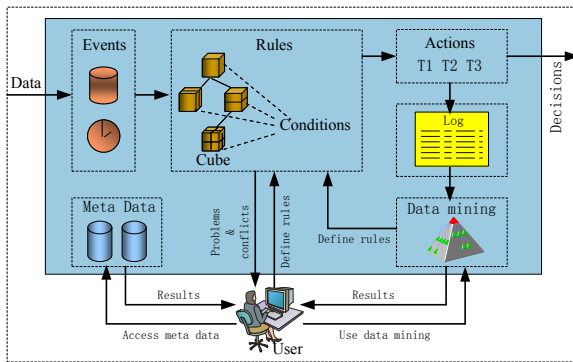
Fig. 1. The architecture of active decision engine in the LADE system

3.2   Cube Using Pattern

Definition 1. ***Cube Using Pattern Matrix:*** *Let $C=\{c_0,c_1,...,c_{m-1}\}$ and $I = \bigcup\limits_{j=0}^{n-1} [t_j, t_{j+1})$, where c_i is a cube, m is the number of cubes used by all non-real-time analysis rules, $[t_j, t_{j+1})$ is a unit interval, and n is the number of unit intervals that a day is divided into. Frequent cube matrix is an $m \times n$ matrix $U = (u_{ij})$ such that $u_{ij} = p$, where p is null or a pointer pointing to a link list.*

In Definition 1, the link list, pointed by u_{ij}, is used to store the $RuleInfoID$ of all those non-real-time analysis rules that use cube c_i during time interval $[t_j, t_{j+1})$. Cube using pattern matrix, U, can be stored in memory for the use of performance optimization algorithm, and we can get it from the action log.

3.3   Frequent Cube

Definition 2. ***Frequent Cube Matrix:*** *Let $C=\{c_0,c_1,...,c_{m-1}\}$ and $I = \bigcup\limits_{j=0}^{n-1} [t_j, t_{j+1})$, where c_i is a cube, m is the number of cubes used by real-time analysis rules, $[t_j, t_{j+1})$ is a unit interval, and n is the number of unit intervals that a day is divided into. Frequent cube matrix is an $m \times n$ matrix $A = (a_{ij})$ such that*

$$a_{ij} = \begin{cases} 1 & if\ c_i\ is\ a\ frequent\ cube\ for\ [t_j, t_{j+1}) \\ 0 & otherwise \end{cases}$$

Frequent cube matrix A can be easily maintained in memory to enhance the performance of optimization algorithm. Also it can be easily extended according to our requirements.

4   Performance Optimization with ARPO Algorithm

Performance optimization work is based on the necessary reference information such as rush hour, cube using pattern matrix and frequent cube matrix. System will do the optimization work whenever an analysis rule is initiated. Algorithm 1 shows the process of performance optimization, in which, q_i in the 15th line is used to store the $RuleInfoID$ of all those rules using c_i. When a rule is initiated, if it is a real-time analysis rule, system will generate the required cube for it. Also, if the cube is a frequent cube for the current time interval, it will be materialized for the use of other coming real-time analysis rules, which will be a great help for improving the performance of those rules. If a non-real-time analysis rule L is initiated, system will first judge if it is rush hour now, or if there are other rules using the same cube as L. If either one of the two conditions evaluates to be TRUE, L will be enqueued into a waiting queue q, which accommodates all those rules sharing the same cube. At an appropriate time, the rules in the waiting queue will be dequeued and continue their analysis process. In this way, system may just generate the required cube one time to satisfy all those rules from the same waiting queue, which also greatly improves the system performance.

Algorithm 1. ARPO(A, U, L, S)

    Input : 1: frequent cube matrix A
               2: cube using pattern matrix U
               3: analysis rule L
               4: rush hour set S
    Output: 1: execution result

---

1 **begin**
2   get the time interval $[t_j, t_{j+1})$ to which the current time belongs;
3 $i \leftarrow L.\ CubeID$;
4   **if** $L.\ IsRealTime=TRUE$ **then**
5   generate the cube c_i if not exist;
6   execute L;
7   **if** $A[i][j] = 1$ **then**
8   materialize the cube c_i if it has not been materialized;
9   **else**
10   delete cube c_i;
11   **end**
12   **return** execution result of L;
13   **else**
14   **if** $([t_j, t_{j+1}) \in S)$ **or** $(U[i][j] \neq NULL)$ **then**
15   initialize a waiting queue q_i if not exist;
16   put $L.\ RuleInfoID$ into q_i;
17   **return** q_i;
18   **end**
19   **end**
20 **end**

---

5   Empirical Study

In this section, we report the performance evaluation of our method. The algorithms are implemented with C++. All the experiments were conducted on Intel i7-2600 3.40GHz CPU, 16.0GB memory DELL PC running Windows 7 and Oracle 11g.

In the LADE system, we use the TPC benchmark TPC-H to get the required datasets. We have been running the LADE system for several months. The action log contains three month of data, from which we can get the rush hour set S, cube using pattern matrix U and frequent cube matrix A.

Experiment 1. We design 50 real-time analysis rules which will use 30 cubes all together in the analysis process. We change the value of f, the percent of frequent cubes to the overall 30 cubes, from 10% to 50%. Fig.2 shows the change of t_1/t_2 during this process, where t_1 is the total time cost for the execution of all these 50 rules without optimization work, and t_2 is the time cost with optimization work. We can get from the experiment result that frequent cube plays an important role in decreasing the execution time of real-time analysis rules. ARPO algorithm can make full use of frequent cubes in the process of
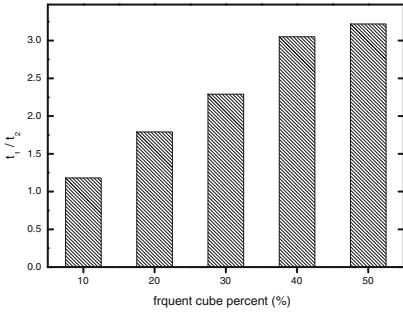
Fig. 2. Time cost ratio
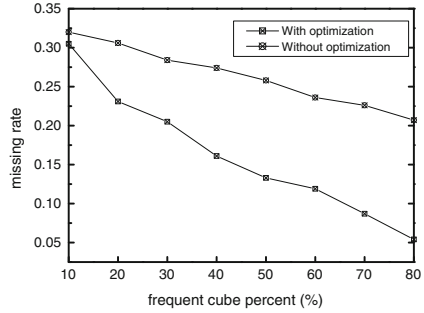
Fig. 3. Missing rate
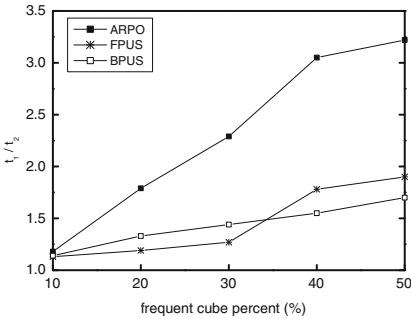
Fig. 4. Time cost ratio comparison
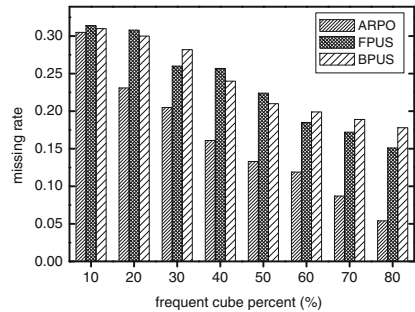
Fig. 5. Missing rate comparison

performance optimization. For example, when f=50%, time cost ratio t_1/t_2 can reach a high value of 3.22.

Experiment 2. If a cube required by a real-time analysis rule is not available right now, which means that it needs to be generated upon the time of requirement, we say that the real-time analysis rule misses this cube. Here we define a new variable $r = a_1/a_2$, where a_1 is the total times that real-time analysis rules miss cubes during a period of time T, and a_2 is the total times that real-time analysis rules require cubes during T. Just as Experiment 1, we change the percent of frequent cubes to the overall 30 cubes from 10% to 80%. From the experiment result in Fig.3, we can get that ARPO algorithm can reduce missing rate greatly. When $f = 10\%$, the values of r before and after optimization are 0.32 and 0.305 respectively. When $f = 80\%$, they are 0.207 and 0.054 respectively.

Experiment 3. Some desirable methods used to deal with the issue of materialized view selection, such as FPUS [6] and BPUS [7], may also help to improve the performance of real-time analysis rules, because they maintain those frequently-

used cubes in the system. Fig.4 shows the time cost ratio of ARPO compared with those of FPUS and BPUS, from which we can get that, ARPO may achieve much better performance than both FPUS and BPUS. As far as ARPO is concerned, the larger the value of f is, the greater the performance improvement is. However, for FPUS and BPUS, the value of t_1/t_2 only change a little when the value of f changes between 10% and 50%. For example, when f=50%, the value of t_1/t_2 for ARPO is 3.22, and for FPUS and BPUS, however, they are only 1.9 and 1.7 respectively. Fig.5 shows the missing rate of ARPO compared with those of FPUS and BPUS. We can observe that, the value of f has much more influence on ARPO than on FPUS and BPUS. In another word, ARPO may take better use of frequent cubes than both FPUS and BPUS.

6   Related Work

Real-time active data warehouse is attracting more and more attention due to the great benefits it may bring to organizations. It can support both strategic and tactic decisions. Analysis rule is a very important part of the real-time data warehouse, which is based on the knowledge developed in the field of active database systems. Paton *et al.* in [8] do a lot of survey work in active database system. This survey presents the fundamental characteristics of active database systems, describes a collection of representative systems within a common framework, considers the consequences for implementations of certain design decisions, and discusses tools for developing active applications.

In [5], Schrefl *et al.* introduce the conception of ECA rules from active database system into the field of data warehouses, and applies the idea of event-condition-action rules (ECA rules) to automate repetitive analysis and decision tasks in data warehouses. The work of an analyst is mimicked by analysis rules, which extend the capabilities of conventional ECA rules in order to support multidimensional analysis and decision making. In [5], the authors also present the architecture of active data warehouse, and describe in detail the knowledge of event model, action model, analysis graph and analysis rules.

In [9], Tho *et al.* combine (1) an existing solution for the continuous data integration and (2) the known approach of active data warehousing by introducing protocols that enable the correct collaboration between the two. Therefore, analysis rules can operate on real-time data.

Up to date, the research work in the field of real-time active data warehouses mostly focuses on the aspects such as the architecture of real-time active data warehouse [9], real-time data integration, real-time data modeling, time consistency [10], query optimization, scalability, real-time alerting, and so on.

7   Discussion and Conclusion

In this paper, we focus on the performance optimization of analysis rules in real-time active data warehouses. The LADE system is designed to get all the reference information required by optimization work, and a new algorithm, called

ARPO, is proposed to carry out the optimization work based on the reference information. Extensive experiments show that our method can effectively improve the system performance of analysis rules.

References

1. Thalhammer, T., Schrefl, M., Mohania, M.: Active Data Warehouses: complementing OLAP with Analysis Rules. Data and Knowledge Engineering 39, 241–269 (2001)
2. Chen, L., Rahayu, J.W., Taniar, D.: Towards Near Real-Time Data Warehousing. In: 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 1150–1157. IEEE press, New York (2010)
3. Polyzotis, N., Skiadopoulos, S., Vassiliadis, P., Simitsis, A., Frantzell, N.: Supporting Streaming Updates in an Active Data Warehouse. In: 23rd International Conference on Data Engineering, pp. 476–485. IEEE Press, New York (2007)
4. Lin, Z.Y., Lai, Y.X., Lin, C., Xie, Y., Zou, Q.: Maintaining Internal Consistency of Report for Real-Time OLAP with Layer-Based View. In: Du, X., Fan, W., Wang, J., Peng, Z., Sharaf, M.A. (eds.) APWeb 2011. LNCS, vol. 6612, pp. 143–154. Springer, Heidelberg (2011)
5. Schrefl, M., Thalhammer, T.: On Making Data Warehouses Active. In: Kambayashi, Y., Mohania, M., Tjoa, A.M. (eds.) DaWaK 2000. LNCS, vol. 1874, pp. 34–46. Springer, Heidelberg (2000)
6. Tan, H.X., Zhou, L.X.: Dynamic selection of materialized views of multidimensional data. Journal of Software 13(6), 1090–1096 (2002)
7. Harinarayan, V., Rajaraman, A., Ullman, J.D.: Implementing data cubes efficiently. In: ACM SIGMOD 1996 International Conference on Management of Data, pp. 205–216. ACM Press, New York (1996)
8. Paton, N.W., Diaz, O.: Active Database Systems. ACM Computing Surveys 31(1), 63–103 (1999)
9. Tho, M.N., Tjoa, A.M.: Zero-Latency Data Warehousing for Heterogeneous Data Sources and Continuous Data Streams. In: 5th International Conference on Information Integrationand Web-based Applications Services, pp. 55–64. Austrian Computer Society, Vienna (2003)
10. Bruckner, R.M., Tjoa, A.M.: Capturing Delays and Valid Times in Data Warehouses-Towards Timely Consistent Analyses. Journal of Intelligent Information Systems 19(2), 169–190 (2002)

Efficient Retrieval of Similar Workflow Models Based on Behavior

Tao Jin[1,2], Jianmin Wang[2], and Lijie Wen[2]

[1] Department of Computer Science and Technology, Tsinghua University, China
[2] School of Software, Tsinghua University, China
{jint05,wenlj00}@mails.thu.edu.cn,jimwang@tsinghua.edu.cn

Abstract. With the workflow technology being more widely used, there are more and more workflow models. How to retrieve the similar models efficiently from a large model repository is challenging. Since dynamic behavior is the essential characteristic of workflow models, we measure the similarity between models based on their behavior. Since the number of models is large, the efficiency of similarity retrieval is very important. To improve the efficiency of similarity retrieval based on behavior, we propose a more efficient algorithm for similarity calculation and use an index named TARIndex for query processing. To make our approach more applicable, we consider the semantic similarity between labels. Analysis and experiments show that our approach is efficient.

1 Introduction

The wide use of workflow technology results in a large number of workflow models. How to manage such a large number of models is challenging, among which the similarity retrieval is a basic function. For example, China CNR Corporation Limited is a newly regrouped company which has more than 20 subsidiary companies. Before the group company was established, most of these subsidiary companies independently deployed their information systems with a total of more than 200,000 workflow models. Now, CNR needs to integrate these information systems. How to group or cluster these models efficiently is a problem. Efficient retrieval of similar workflow models is helpful in tackling this challenge.

The problem to be solved in this paper can be described as follows. Given a query condition model q, quickly find all the models (notated as S) in the workflow model repository R satisfying that, for every model m in S, the similarity between m and q must be greater than or equal to a specific threshold θ.

The behavior of a workflow model describes how the business process runs, and a business process can be represented as topologically different graphs, so dynamic behavior is the essential characteristic of workflow models. We measure the similarity between models based on their task adjacency relations, which is the topic of [1]. Our contributions in this paper can be summarized as follows.

– We propose a more efficient algorithm to compute the task adjacency relations. The problem of state explosion in [1] has been solved.

- We use an index named *TARIndex* to speed up the query processing.
- We consider the semantic similarity between labels to make our approach more applicable.
- We implement our approach in the BeehiveZ environment and do some experiments to evaluate our approach.

2   Preliminaries

We use sound SWF-nets to represent workflow models in this paper, because WF-nets that are not SWF-nets are difficult to understand and should be avoided, and there are some errors in unsound workflow models so that they should be avoided. The definitions of *Petri net*, *WF-net*, *sound* and *SWF-net* can be found in [2]. In [1], the similarity measure between workflow models based on transition adjacency relation (TAR) set was proposed, which is used to calculate the similarity between sound SWF-nets based on behavior.

Definition 1 (TARS and TARS similarity). *Let FS be the full firing sequences of a sound SWF-net $N = (P, T, F)$. For $a, b \in T$, $\langle a, b \rangle$ is a transition adjacency relation (TAR) of N if and only if there is a trace $\sigma = t_1 t_2 t_3 ... t_n$ and $i \in \{1, 2, ..., n-1\}$ such that $\sigma \in FS$, $t_i = a$ and $t_{i+1} = b$. The complete TARs in FS is denoted as $TARS$. Let N_1 and N_2 be two sound SWF-nets. Let $TARS_1$, and $TARS_2$ be their TARS respectively. Then, the similarity between N_1 and N_2 can be calculated by using Equation 1.*

$$tarSim(N_1, N_2) = \frac{|TARS_1 \cap TARS_2|}{|TARS_1 \cup TARS_2|}. \tag{1}$$

3   A New Algorithm for TARS Computation

To solve the problem of state explosion in [1], we compute the TARS of a sound SWF-net based on its complete prefix unfolding [3] instead of on its reachability graph in this section. The procedure is described as Algorithm 1. The computation can be divided into three stages.

Firstly, we traverse all the places in the complete prefix unfolding and obtain some TARs, as Lines 1-4 show. For the source place i, for every $t \in i\bullet$, we add $\langle null, t \rangle$ to the TARS, which is not shown in Algorithm 1. By doing this, we can deal with the transitions that the source place is the sole input place and the sink place is the sole output place.

Secondly, we complement some TARs for the cut-off transitions, as Lines 5-6 show. For every cut-off transition, we get the corresponding transition. Then we pair the cut-off transition and the successors of the corresponding transition to obtain some new TARs.

Thirdly, we can shuffle all the transitions in parallel in the complete prefix unfolding to finish the TARS computation, as Lines 7-10 show. In the first two stages, we obtain the TARs through the places existing between transitions.

Algorithm 1. Compute TARS

---

input : A sound SWF-net $W = (P_W, T_W, F_W)$, its complete prefix unfolding
$U = (P, T, F)$, and the mapping function $l_W : T \to T_W$
output: the TARS of the corresponding sound SWF-net

```
   // traverse the complete prefix unfolding and get some TARs
1  foreach p ∈ P do
2  |   foreach tpre ∈ •p do
3  |   |   foreach tpost ∈ p• do
4  |   |   |   add ⟨tpre, tpost⟩ to TARS;

   // deal with the cut-off transitions and add some TARs
```
5 **foreach** $t_i, t_j, t_k \in T$ such that t_i is a cut-off transition in U, t_j is not a cut-off
transition, $Mark(\lceil t_i \rceil) = Mark(\lceil t_j \rceil)$, and $l_W(t_i) \bullet \cap l_W(t_j) \bullet \cap \bullet l_W(t_k) \neq \emptyset$ **do**
6 | add $\langle t_i, t_k \rangle$ to TARS;
```
   // shuffle all the transitions in parallel to complement TARS
7  foreach tᵢ ∈ T do
8  |   foreach tⱼ ∈ T do
9  |   |   if tᵢ and tⱼ are in concurrency relations then
10 |   |   |   add ⟨tᵢ, tⱼ⟩ and ⟨tⱼ, tᵢ⟩ to TARS;

11 return TARS;
```

However, for transitions which can be executed parallelly, there are no places connecting them. For each pair of transitions which can be executed parallelly, we can get two TARs, as Line 10 shows. Finally, Line 11 returns the TARS. To get all pairs of transitions in concurrency relations, we use the algorithm presented in [4] to compute the ordering relations directly.

4 Index Construction and Query Processing

We use an index named *TARIndex* in this section to work as a filter. We only need to compute the similarity between the query condition model and every candidate model obtained with the help of *TARIndex*, so the query efficiency can be improved. Moreover, the computation of TARS for the models in the repository is completed during the *TARIndex* construction, so during query processing the time is saved.

4.1 Index Construction

TARIndex sets up the relation between TARs and models, and it has two parts. One part is a forward index (notated as *FI*), The items indexed in *FI* are like $(m, TARS)$, in which, m is denoted as a model, and *TARS* is the set of task adjacency relations of the corresponding model. The other part is an inverted index (notated as *II*). The items indexed in *II* are like $(TAR, TAR.list)$, in

which, *TAR* is denoted as a task adjacency relation, and *TAR.list* is denoted as a set of models where the corresponding *TAR* appears.

Algorithm 2. Add a model to *TARIndex*

---

input: a model m represented as a sound SWF-net

1 TARS = computeTARS(m);
2 **foreach** tar *in* TARS **do**
3   FI.add(m,tar);
4   II.add(tar,m);

Based on the Algorithm 1, the *TARIndex* can be constructed as described in Algorithm 2. Line 1 computes the TARS of the given sound SWF-net, and then, Lines 2-4 traverse every TAR and update the forward index (*FI*) in Line 3 and the inverted index(*II*) in Line 4. From Algorithm 2, we can see that when a new model is added to the repository, the index can be updated incrementally.

4.2 Query Processing

Based on *TARIndex*, the query processing can be divided into two stages, namely, the filtering stage and the refinement stage. The procedure is described as Algorithm 3. Line 1 obtains the TARS of the query condition model. Lines 2-3 obtain the set of candidate models where at least one TAR of query condition model appears. Lines 4-8 calculate the similarity between every candidate model and the query condition model. Line 5 uses the forward index (*FI*) to obtain the TARS of a candidate model. Function `tarSim` in Line 6 calculates the similarity between models according to Equation 1. If the similarity is less than the

---

Algorithm 3. Retrieve the similar models

---

input : a query condition model q, and the model similarity threshold θ
output: all the models satisfying requirement

   `// filtering stage`
1 qTARS = computeTARS(q);
2 **foreach** tar *in* qTARS **do**
3   ret.add(II.getModelSet(tar));
   `// refinement stage`
4 **foreach** c *in* ret **do**
5   mTARS = FI.getTARS(c);
6   similarity = tarSim(qTARS,mTARS);
7   **if** similarity $< \theta$ **then**
8   ret.remove(c);

9 **return** ret;

---

specified threshold, the candidate model is removed, as Lines 7-8 show. Finally, Line 9 returns all the remaining models satisfying the requirements. Here, we can see that the model similarity threshold has no effect on the query efficiency.

4.3   Dealing with Semantic Similarity between Labels

Business analysts may choose different names for identical tasks in process models. In order to recognize the identical tasks we extend our approach with a notion of semantic similarity for labels. Tasks with their label similarity not less than a specified threshold are considered to be identical. The semantic similarity measurement between labels, the label index construction and also the query processing with label similarity considered are similar to the work in [4].

5   Tool Support and Evaluation

To evaluate our approach, we implemented it in our system named BeehiveZ[1]. During our experiments, we used a computer with Intel(R) Core(TM)2 Duo CPU E8400 @3.00GHz and 3GB memory. This computer ran Windows XP Professional SP3 and JDK6. The heap memory for JVM was configured as 1GB.

5.1   TARS Computing Performance Comparison

We compute TARS of business process models based on complete prefix unfolding in this paper. Compared to the method based on reachability graph, the performance can be improved greatly, especially when there are many tasks in parallel. In this situation, the construction of complete prefix unfolding has a linear time to the number of tasks, while the construction of reachability graph has an exponential time to the number of tasks. The dominating factor for TARS computation complexity is the construction of complete prefix unfolding or reachability graph, so our new method for TARS computation is superior. This conclusion can be drawn from Table 1, where "n" means how many tasks are in parallel, "CP" means our method based on complete prefix unfolding, and "RG" means the method based on reachability graph.

5.2   Experiments on a Synthetic Data Set

In this section, we conduct experiments on a synthetic data set to show that the performance is improved by using *TARIndex*. All the models in our experiments are generated automatically using the rules from [5]. We generated models with the maximum number of transitions as 50. Because all the labels are strings generated automatically, we disabled the use of label similarity.

The query time comparison between query using *TARIndex* and query without index can be found in Fig. 1(a). We can find that the use of *TARIndex* improves the retrieval efficiency greatly. With more models added to the repository,

---

[1] BeehiveZ can be downloaded from http://code.google.com/p/beehivez/

Table 1. Performance comparison on TARS computing for models with many tasks in parallel

n	3	7	11	15	19	38	76
CP	0ms[*]	0ms[*]	3ms	3ms	3ms	9ms	38ms
RG	0ms[*]	31ms	1175ms	38900ms	-[+]	-[+]	-[+]

[*] It cost less than 1 millisecond.
[+] It is not computable because of the error of out-of-memory.

(a) query time comparison

(b) query time

(c) index construction time

(d) storage size
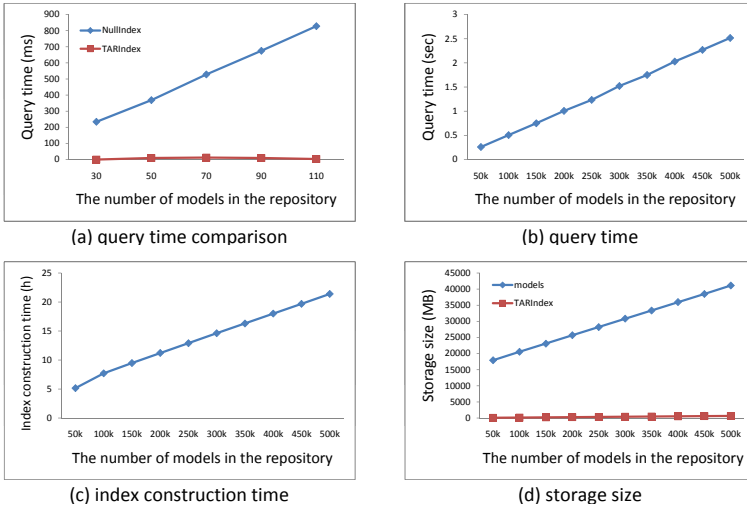
Fig. 1. The performance of *TARIndex*

it is more and more time-consuming. But the retrieval time is still acceptable. The change of retrieval efficiency can be found in Fig. 1(b). Since for different query condition models, the change of query time is similar, we only show the query time on the query condition model with 26 transitions here. The index construction time can be found in Fig. 1(c). It shows the accumulated time for index construction from scratch. Since our index can be constructed incrementally, when new models are added to the repository, the index can be updated immediately, the time for index updating is very short. The storage size of index can be found in Fig. 1(d). The storage size of index is less than 2% of the models. Now, we can draw a conclusion that *TARIndex* can improve the efficiency of similarity retrieval greatly at little cost.

5.3 Experiments on SAP Reference Models

We did experiments on SAP Reference models to show how label similarity affects the query efficiency. First, all these models (had been transformed into

Petri nets) were added to the repository, and then both the *TARIndex* and the *label index* were built, then every model was used as a query on the repository with different label similarity thresholds. The change of query time with different label similarity thresholds can be found in Table 2. From this table, we can see that with the label similarity considered there are more result models and the query time increases, and with the label similarity threshold increases the size of result set decreases and the query time decreases.

Table 2. Query time (ms) with different label similarity thresholds

	disabled	0.5	0.6	0.7	0.8	0.9	1.0
Min	0*	0*	0*	0*	0*	0*	0*
Max	31	2293.4	1634.6	1634.4	1618.8	1637.4	1619
Average	3.11	77.48	66.47	65.27	64.88	64.76	64.48
St. Dev.	3.99	199.93	163.67	160.48	160.55	161.44	159.19

  * It cost less than 1 millisecond.

6   Related Work

There are already some works on business process model query, such as the work in [6,7,8,9,10]. There are also many works in graph search area, such as the work mentioned in [11]. But all the above works only focus on graph structure while we focus on behavior (semantics) in this paper. A good overview on workflow similarity search was given in [12]. All the works mentioned in [12] have not considered using index to improve the efficiency, so that they must compute the similarity between the query condition model and every model in the repository. It is too much time-consuming while we solve this problem with the use of indexes. [4] deals with the exact query based on behavior while we deal with similarity query based on behavior in this paper.

7   Conclusion and Future Work

In this paper we focus on improving the efficiency of similar workflow model query based on behavior. To compute the similarity between models based on behavior, we use the notion of TARS. To compute TARS efficiently, we propose an algorithm based on complete prefix unfolding. Compared to the algorithm based on reachability graph, our new algorithm solves the problem of state explosion so that the efficiency can be improved. To improve the query efficiency more, we use an index named *TARIndex* to support query processing. *TARIndex* works as a filter, so that we only need to compute the similarity between the query condition model and every model in the candidate set which is obtained with the use of *TARIndex*. The size of candidate set is always much smaller than the size of the repository, that is why the query efficiency can be improved. Moreover, the TARS computation of models in the repository is completed during the

TARIndex construction, so during the query processing the time is saved. To make our approach more applicable, we consider the semantic similarity between labels, and the query efficiency is still high with the use of the *label index*.

However, we only consider the tasks and the execution order between tasks when we measure the similarity between business process models. We will also consider the data processing and the resource allocation in the future.

References

1. Zha, H., Wang, J., Wen, L., Wang, C., Sun, J.: A Workflow Net Similarity Measure based on Transition Adjacency Relations. Computers in Industry 61(5), 463–471 (2010)
2. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Trans. Knowl. Data Eng. 16(9), 1128–1142 (2004)
3. Esparza, J., Römer, S., Vogler, W.: An Improvement of McMillan's Unfolding Algorithm. Formal Methods in System Design 20(3), 285–310 (2002)
4. Jin, T., Wang, J., Wen, L.: Querying business process models based on semantics. In: Yu, J.X., Kim, M.H., Unland, R. (eds.) DASFAA 2011, Part II. LNCS, vol. 6588, pp. 164–178. Springer, Heidelberg (2011)
5. Murata, T.: Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE 77(4), 541–580 (1989)
6. Beeri, C., Eyal, A., Kamenkovich, S., Milo, T.: Querying Business Processes. In: VLDB, pp. 343–354 (2006)
7. Scheidegger, C.E., Vo, H.T., Koop, D., Freire, J., Silva, C.T.: Querying and Re-Using Workflows with VisTrails. In: SIGMOD Conference, pp. 1251–1254 (2008)
8. Shao, Q., Sun, P., Chen, Y.: WISE: A Workflow Information Search Engine. In: ICDE, pp. 1491–1494 (2009)
9. Jin, T., Wang, J., Wu, N., La Rosa, M., ter Hofstede, A.H.M.: Efficient and Accurate Retrieval of Business Process Models through Indexing. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010, Part I. LNCS, vol. 6426, pp. 402–409. Springer, Heidelberg (2010)
10. Jin, T., Wang, J., Wen, L.: Efficient Retrieval of Similar Business Process Models Based on Structure. In: Meersman, R., Dillon, T., Herrero, P., Kumar, A., Reichert, M., Qing, L., Ooi, B.-C., Damiani, E., Schmidt, D.C., White, J., Hauswirth, M., Hitzler, P., Mohania, M. (eds.) OTM 2011, Part I. LNCS, vol. 7044, pp. 56–63. Springer, Heidelberg (2011)
11. Ke, Y., Cheng, J., Yu, J.X.: Querying Large Graph Databases. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010, Part II. LNCS, vol. 5982, pp. 487–488. Springer, Heidelberg (2010)
12. Dumas, M., García-Bañuelos, L., Dijkman, R.M.: Similarity Search of Business Process Models. IEEE Data Eng. Bull. 32(3), 23–28 (2009)

Scalable Subspace Logistic Regression Models for High Dimensional Data

Shuang Wang, Xiaojun Chen, Joshua Zhexue Huang, and Shengzhong Feng

Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
Shenzhen 518055, China
{shuang.wang,xj.chen,zx.huang,sz.feng}@siat.ac.cn

Abstract. Although massive, high dimensional data in the real world provide more information for logistic regression classification, yet it also means a huge challenge for us to build models accurately and efficiently. In this paper, we propose a scalable subspace logistic regression algorithm. It can be viewed as an advanced classification algorithm based on a random subspace sampling method and the traditional logistic regression algorithm, aiming to effectively deal with massive, high dimensional data. Our algorithm is particularly suitable for distributed computing environment, which we have proved, and it is implemented on Hadoop platform with MapReduce programming framework in practice. We have done several experiments using real and synthetic datasets and demonstrated better performance of our algorithm in comparison with other logistic regression algorithms.

Keywords: high dimensional data, ensemble learning, logistic regression, MapReduce.

1   Introduction

Logistic regression is useful for two-class classification, which is widely applied in many areas, such as document classification and natural language processing [1]. However, the data with millions of records and thousands of features present a huge challenge to the traditional logistic regression algorithm. On the one hand, it is difficult to build a logistic regression model from such massive data with serial logistic regression algorithms running on single machines. On the other hand, traditional logistic regression algorithms are not capable of building accurate models from extremely high dimensional data with thousands of features. Thus, new technologies are required to improve the traditional logistic regression algorithm for massive, high dimensional data.

There are many algorithms for training logistic regression models [2], for instance, iterative scaling, nonlinear conjugate gradient, quasi Newton and so on. All the optimization algorithms are iterative procedures. The recently proposed TR-IRLS [3] is a more effective optimization algorithm. This algorithm is a simple modification of iteratively re-weighted least squares(IRLS) that mimics truncated Newton methods to effectively train models, and it can quickly train a

logistic regression model for high dimensional data. However, the models trained by TR-IRLS contain all the attributes, without considering the irrelevant and redundant information included in high dimensional data, which may greatly degrade the performance of learned models [4]. Therefore, it has inspired several feature selection techniques to pre-process high dimensional data and feature selection algorithms to reduce the dimensionality to enable learning algorithms to work effectively on high dimensional data [5]. However, they are time consuming and also possibly delete some critical features. Focusing on high dimensional data, researchers have proposed a method applying the Random Forest principles to MultiNomial Logit [5], which can solve the time consuming problems of feature selection and enhance the accuracy of the final classification model. However, when dealing with massive data, the running time is too long to bear. Therefore, it does not have good scalability.

In this paper, we present a scalable subspace logistic regression algorithm (SSLR) in order to effectively deal with high dimensional, massive data. To achieve the goal of dealing with high dimensional data, we also apply Random Forest principles to randomly select multiple subspaces to train multiple logistic regression models. Besides, the method costs less time than the traditional feature selection algorithms. For better prediction, we ensemble these models with the voting method to get the best result. Thus, through the above procedure, we can more accurately classify the data. To achieve the goal of dealing with massive data, our algorithm is designed particularly suitable for distributed computing environment. Massive data can be distributed into separate nodes, and our algorithm can train multiple models on different nodes in parallel. In practice, we implement our algorithm using MapReduce [6] programming framework. We have conducted experiments on clusters with 30 nodes installed with Hadoop [7,8,9]. The results have shown that our algorithm outperformed traditional algorithms in classification accuracy and MapReduce's distributed computing ability endows our model good scalability.

The rest of the paper is organized as follows. Section 2 gives an introduction to logistic regression model and the algorithm we use to train single logistic regression models. Section 3 describes our algorithm in details. In Section 4, we show the implementation details of our algorithm with MapReduce. Experimental results on both synthetic data and real data are presented in Section 5. We draw conclusions in Section 6.

2   Preliminary

2.1   Logistic Regression Model

Logistic regression model is a well-known two-class classification method. Logistic regression function returns the probability of one instance in positive class. It is written as:

$$f(\mathbf{x}, \beta) = \frac{exp(\beta\_0 + \beta\_1 \mathbf{x}\_1 + \ldots \beta\_k \mathbf{x}\_k)}{1 + exp(\beta\_0 + \beta\_1 \mathbf{x}\_1 + \ldots \beta\_k \mathbf{x}\_k)} \qquad 2.1$$

where $\beta = (\beta\_0, \beta\_1, \beta\_2, ..., \beta\_k)^T$ is the logistic function parameters, $\mathbf{x}\_i$ is the i-th feature value of $\mathbf{x}$, which is the vector $(\mathbf{x}\_1, \mathbf{x}\_2, \mathbf{x}\_3, ..., \mathbf{x}\_M)$(M is the the number of features of $\mathbf{x}$). To compute the parameters for the logistic regression function, we know that the probability of one instance in the positive class is f($\mathbf{x}$,β), and the probability in the negative class is 1 - f($\mathbf{x}$,β). The probability of y given $\mathbf{x}$ and β can be written as the following equation:

$$P(y|\mathbf{x}, \beta) = \begin{cases} f(\mathbf{x}, \beta) & \text{if } y = 1, \\ 1 - f(\mathbf{x}, \beta) & \text{if } y = 0. \end{cases}$$

2.2

From the expression, the likelihood and the log-likelihood of the training data X under the logistic regression model with parameters β can be written as:

$$L(X, \mathbf{y}, \beta) = \prod_{i=1}^{N} f(\mathbf{x}_i, \beta)^{\mathbf{y_i}}(1 - f(\mathbf{x}_i, \beta))^{1-\mathbf{y_i}}$$

2.3

$$lnL(X, \mathbf{y}, \beta) = \sum_{i=1}^{N}(\mathbf{y}_i ln(f(\mathbf{x}_i, \beta)) + (1 - \mathbf{y}_i)ln(1 - f(\mathbf{x}_i, \beta)))$$

2.4

where $\mathbf{x}_i$ is the i-th instance of X, $\mathbf{y_i}$ is the classification result of the i-th instance, and N is the number of instances.

2.2   TR-IRLS

There are several algorithms to solve the maximum-likelihood logistic regression problem. In our work, we use TR-IRLS proposed by Paul R. Komarek [3] to train the logistic regression model. To compute the maximum of the log-likelihood, firstly, we differentiate the function:

$$\frac{\partial}{\partial \beta}lnL(X, \mathbf{y}, \beta) = X^T(\mathbf{y} - f(X, \beta))$$

2.5

Finding the maximum likelihood equation is equivalent to finding the zeros of the above function. The Newton-Raphson algorithm [10] can be used to find the optimum. The Newton-Raphson update formula can be written as:

$$\beta^{update} = \beta + (X^TDX)^{-1}X^T(\mathbf{y} - f(X, \beta))$$

2.6

where D is the diagonal matrix with $d_{ii} = f(\mathbf{x_i}, \beta)(1 - f(\mathbf{x_i}, \beta)$. Since $\beta = (X^TDX)^{-1}X^TDX\beta$, the above equation can be rewritten as:

$$\beta^{update} = (X^TDX)^{-1}X^T(DX\beta + (\mathbf{y} - f(X, \beta))) = (X^TDX)^{-1}X^TDz$$

2.7

where $z = X\beta + D^{-1}(\mathbf{y} - f(X, \beta))$. The above equation can be further written as

$$(X^TDX)\beta^{update} = X^TDz$$

2.8

The TR-IRLS method uses the conjugate gradient method to solve β^{update} in Equation 2.8 [11].

3   Scalable Subspace Logistic Regression

3.1   Generating Multiple Subspaces

Let F = $\{f_1, f_2, ..., f_m\}$ be the set of all features in the training data. We randomly select W features from F creating a subspace F_i, and repeat this operation K times. Then, we can create subspace set S = $\{F_1, F_2, F_3, ..., F_K\}$. Because the features selected are random, it is possible that the different subspaces overlap. We set a condition to make sure that the features in different subspaces will not be all the same. For a given feature space of M dimensions and a fixed number of selected features W, there are C_M^W selections that can be constructed. Since K $\ll C_M^W$, so we use the random method to explore the possibilities in a convenient way [12].

3.2   Building Multiple Logistic Regression Models

After generating the multiple subspaces, we train multiple models using the following steps:

(a) Partition the training data into P nodes in a distributed system. The data sets on different nodes are denoted as TP = $\{T_1, T_2, T_3, ..., T_P\}$, the classification outcomes of data on P nodes are notated as YP = $\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, ..., \mathbf{y}_P\}$. We distribute S and the initial parameters $\{\beta^1, \beta^2, \beta^3, ..., \beta^K\}$ of models to every node.

(b) In each node i, project subspaces to corresponding feature values in TP, and create training datasets for different subspaces TPS = $\{T_{i1}, T_{i2}, T_{i3}, ..., T_{iK}\}$. The outcome of each instance in TPS is the same as original outcome of each instance in T.

(c) In each node i, from the initial logistic regression function parameters, compute different values of $T_{ij}^T D_{ij} T_{ij}$ and $T_{ij} D_{ij} z_{ij}$ for different models. Note that D_{ij} is a diagonal matrix, with each diagonal element equal to $f(x, \beta^j)$ $(1 - f(x, \beta^j))$(x is the instance in T_{ij}). z_{ij} is the vector equal to $T_{ij}\beta^j + D_{ij}^{-1}(\mathbf{y_i} - f(T_{ij}, \beta^j))$.

(d) In the next step, we sum up all the values of $T_{ij}^T D_{ij} T_{ij}$ and $T_{ij} D_{ij} z_{ij}$ from different nodes, and use

$$T_{ij}^T D_{ij} T_{ij} \beta^j = T_{ij}^T D_{ij} z_{ij}$$

to compute the logistic regression parameters β^j.

(e) Update the parameters of models, redistribute them to separate nodes, and repeat the operations (c),(d) until the parameters converge. Finally, generate the set of K logistic regression models L = $\{LR_1, LR_2, LR_3, ..., LR_K\}$. We further prove the following theorem, which makes sure that logistic regression function parameters computed in the distributed computing environment are equivalent to the parameters computed in a single machine environment. Note that TS = $\{T_1, T_2, T_3, ..., T_K\}$ as the created datasets by projecting subspaces to feature values in TS, D_j , z_j are computed with the same method as computing D_{ij}, z_{ij} .

Theorem 1. *The solution of equation $T_{ij}^T D_{ij} T_{ij} \beta^A = T_{ij}^T D_{ij} T_{ij}$ equals to the solution of equation $T_j^T D_j T_j \beta^B = T_j^T D_j z_j$ where $T_j \in TS$, i.e., $\beta^A = \beta^B$*

Proof. T_j^T, D_j, T_j can be written as follows:

$$T_j^T = \left(T_{1j}^T \; T_{2j}^T \; ..... \; T_{Pj}^T \right) \; D_j = \begin{pmatrix} D_{1j} & & & \\ & D_{2j} & & \mathbf{0} \\ & & ... & \\ \mathbf{0} & & ... & \\ & & & D_{Pj} \end{pmatrix} \; T_j = \begin{pmatrix} T_{1j} \\ T_{2j} \\ \vdots \\ T_{Pj} \end{pmatrix} \quad (1)$$

According to the partitioned matrix multiplication theorem, we can compute the multiplication of above three matrices as follows:

$$T_j^T * D_j * T_j = \left(T_{1j}^T * D_{1j} * T_{1j} + T_{2j}^T * D_{2j} * T_{2j} + ... + T_{Pj}^T * D_{Pj} * T_{Pj} \right) = \sum_{i=1}^{P} T_{ij}^T D_{ij} T_{ij} \quad (2)$$

Similarly, we can prove that $\sum_{i=1}^{P} T_{ij} D_{ij} z_{ij} = T_j^T D_j z_j$. Therefore, we have $\beta^A = \beta^B$.

For classification and prediction, we integrate the sperate models to predict the result of testing data sets. We get a prediction result for each instance with separate models. Then, we use the voting method [13] to predict the final result.

4   Implementation with MapReduce

To handle massive data, we use a popular distributed programming model MapReduce to implement our algorithm [6]. We first partition the training data according to rows, and allow each mapper to compute the intermediate values of matrices and vectors on one object. After this step, mappers will pass the keys and values to the reducers. In the next step, the reducers perform simple matrix summation and vector summation for each subspace to generate the final result of matrices and vectors.

We use a driver program to control mappers and reducers. The driver starts with initialization of parameters set B of multiple logistic regression functions and the index set C for noting the convergence of parameters. For example, C = {0,0,0,...,0} stands for parameters of all the logistic regression functions not convergent. C = {1,0,1,0,...,0} indicates parameters for the first model and the third model convergent. After that, the driver generates subspace set S and transmits B, C, S information to mappers. After the result matrices and vectors are output by reducers, the driver uses the matrices and vectors to calculate new parameters for different models and determines whether or not to restart map and reduce tasks according to the convergency of models. If the parameters of all the models are convergent, the driver stops dispatching tasks. If some models' parameters are not convergent, the driver restarts the map and reduce tasks to recalculate the parameters until all parameters converge. The pseudo code of the driver, map and reduce procedures are described in Algorithms 1, 2, 3.

Algorithm 1. Driver Procedure

Input: Subspace size W, the number of subspaces K, C = {0,0,0,...,0}.
Output: Logistic Regression models set LR = $\{LR_1, LR_2, LR_3, ..., LR_K\}$
 1: Generate subspace set S, initial B
 2: **while** C has zero elements **do**
 3:    Configure Map Reduce Task
 4:    Transmit S, B, C to mappers and reducers
 5:    Dispatch mappers and reducers
 6:    Output result
 7:    Calculate parameters
 8:    Update C
 9: **end while**

Algorithm 2. Map Procedure

Input: Training data T = $\{(x_i, y_i)$ (1≤i≤N)}, subspace set S = $\{F_1, F_2, F_3, ..., F_K\}$,
   logistic regression function parameters set B = $\{\beta^1, \beta^2, \beta^3, ..., \beta^K\}$.
Output: Intermediate data T_d
 1: **for** each subspace $F_i \in S$ **do**
 2:    **if** the i-th logistic regression model is not converged **then**
 3:       In x, project F_i to x_i
 4:       $D_i = f(x_i, \beta^i)(1 - f(x_i, \beta^i))$, $z_i = x_i\beta^i + D_i^{-1}(y - f(x_i, \beta^i))$
 5:       $IM_1 = x_i^T D_i x_i$, $IM_2 = x_i^T D_i z_i$
 6:       key = (i,j,k), value = $(IM_1[j][k], IM_2[j])$(0≤j,k≤W-1))
 7:       store(key,value) in the intermediate data T_d
 8:       $T_d = T_d(key, value)$;
 9:    **end if**
10: **end for**

Algorithm 3. Reduce Procedure

Input: Intermediate data T_d
Output: Matrix set A = $\{A_1, A_2, A_3, ..., A_K\}$, B = $\{B_1, B_2, B_3, ..., B_K\}$ for K models

 1: **for** i = 1 to K **do**
 2:    **if** the i-th logistic regression model is not converged **then**
 3:       set Matrix A = **0**, Vector B = **0**
 4:       **for** each (key,value) in T_d **do**
 5:          **for** j = 0 to W-1 **do**
 6:             **for** k = 0 to W-1 **do**
 7:                if(key == (i,j,k)) then
 8:                $A_i[j][k]$ = A[j][k] + $IM_1[j][k]$;
 9:             **end for**
10:             $B_i[j]$ = B[j] + $IM_2[j]$;
11:          **end for**
12:       **end for**
13:    **end if**
14: **end for**

5 Experiments

In this section, we show the experimental results of SSLR on real and synthetic datasets. We show the accuracy results comparing with other algorithms, and the scalability of our algorithm in dealing with large scale data. 30 machines were used in the experimental environment, each having eight 2.13GHz Intel(R) Xeon(R) processors and 25G memory, running CentOS Linux operating system. We used the latest version of Hadoop, hadoop-0.20.2, to form a MapReduce programming environment.

Two types of data sets were used in our experiments. The first type of data was the real data, used to evaluate the accuracy of our models. We have selected six data sets. The data sets were downloaded from UCI repository [14] and the datasets [1] provided in [1]. Because of the unbalance of Donation dataset, i.e., the proportion between the positive outcomes and the negative outcomes is 1:20, we selected instances randomly from the training data to keep the proportion as 4:6. The characteristics of these data sets are described in Table 1.

Table 1. Real data sets

DataSet	class	no. of features	feature type	instance number of training data	instance number of testing data
ARCENE	2	10000	Real	100	100
Hill_Valley	2	100	Real	606	606
Madelon	2	500	Real	2000	600
Donation	2	3848	Real	9595	12183
Giseete	2	5000	Integer	6000	1000
a9a	2	123	Real	32561	16281

The second type of data is the synthetic data. We developed a data generator, which can produce data of various sizes. We generated four data sets using this generator in our experiments, which are shown in Table 2.

Table 2. Synthetic data sets

Data Set	class	no. of features	feature type	instance number of data
D1	2	1,000	Real	200,000
D2	2	1,200	Real	500,000
D3	2	1,400	Real	800,000
D4	2	1,600	Real	1,000,000

5.1 Accuracy

In this part, we evaluate the performance of our algorithm with classification accuracy. For selecting a better number of features and the number of models, we

---

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Table 3. Accuracy comparison

DataSet	SSLR	TR-IRLS	WekaLR
ARCENE	0.845	0.83	0.75
Hill_Valley	0.995	0.612	0.922
Madelon	0.588	0.56	0.58
Donation	0.602	0.51	0.5169
Gisette	0.917	0.98	0.625
a9a	0.821	0.849	0.849

conducted several experiments focusing on different conditions. Figure 1 shows the accuracy curve of our algorithm against the number of models. We notice that the accuracy increases with the increment of the number of models, but the accuracy only changes in a small range when the number of models exceeds a threshold. Figure 2 shows the accuracy curve of our algorithm against subspace size. From Figure 2, we can notice a similar phenomenon as in Figure 1. With the increase of subspace size, the accuracy also increases, but the accuracy only changes in a small degree when the subspace size exceeds a certain value. Finally, we find some critical points in different data sets where the accuracy nearly approaches to a limit value or a small range. Then, we did experiments on that point for many times and took the average of all the experiment results as our final accuracy of our algorithm. In Table 3, we show the accuracy of our algorithm, comparing with TR-IRLS and the logistic regression classifier in Weka [15].

The result shows that when the accuracy of the traditional logistic regression algorithm is low, the accuracy of our model gains a remarkable increase. If the accuracy of the traditional logistic regression algorithm is high, the accuracy of our model is also high but our algorithm uses less features. To sum up, our algorithm has good accuracy in dealing with high dimensional data.

(a) Arcene   (b) Madelon   (c) Hill_Valley

(d) Donation   (e) Gisette   (f) a9a
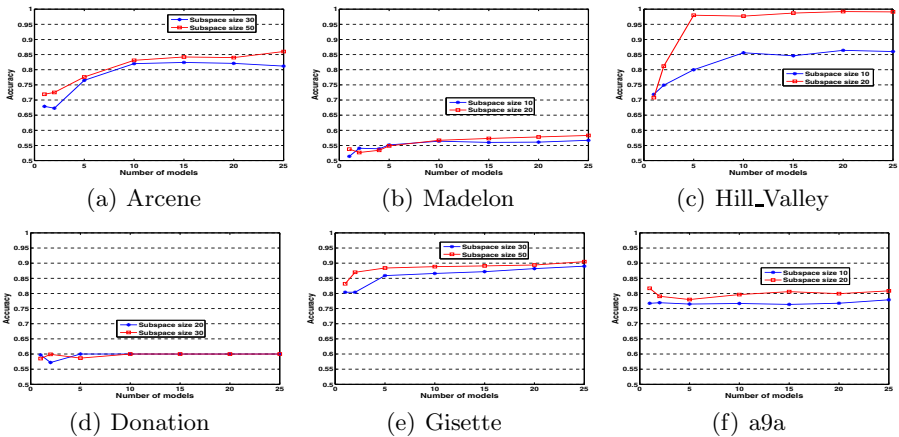
Fig. 1. Accuracy curve against the number of models
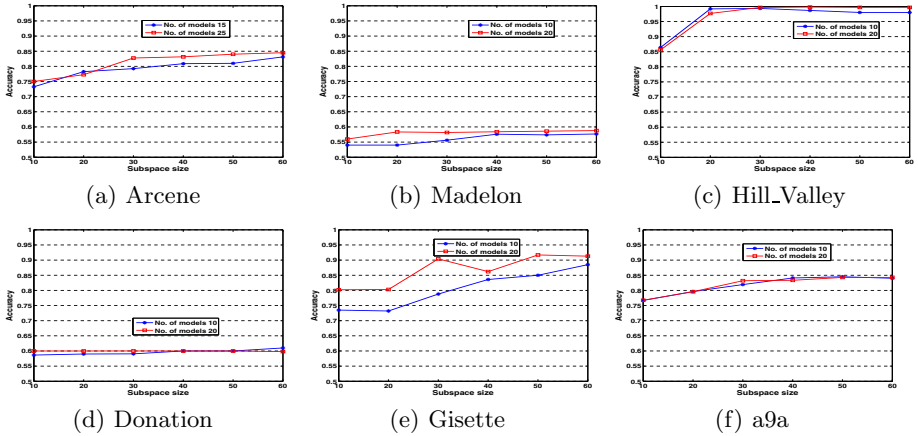
Fig. 2. Accuracy curve against subspace size

5.2   Scalability

In this part, we have carried out two groups of experiments to evaluate the performance of scalability. In the first group of experiments, we fixed the subspace size as 10, the number of subspaces as 10, then we trained models for different sizes of data running on different numbers of nodes. We present the time of our algorithm in dealing with different sizes of data running on 10 nodes, 20 nodes and 30 nodes respectively in the left-chart of Figure 3. In the second group of experiments, we present the time curve of training different numbers of models for different sizes of data running on 30 nodes in the right chart of Figure 3, where the subspace size was fixed to 10. Through the left chart shown in Figure 3, we can see that the running time dropped rapidly as more machines added. This demonstrates that our method can handle large data by adding more machines. Besides, through the right chart shown in Figure 3, we can notice that the larger the data set, the more time it takes to train models when models are added. However, the speed of time increase is not fast. This result demonstrates that SSLR is scalable to the number of models.
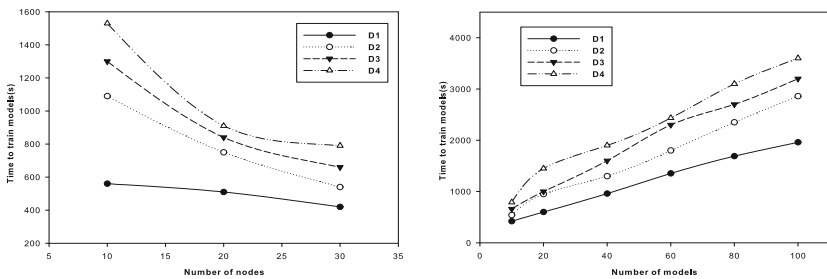
Fig. 3. Scalability tests

6   Conclusions

In this paper, we have proposed a scalable logistic regression algorithm, which is focused on large scale data classification problems. We applied the random subspace method to select multiple subsets of features, then trained different models in a distributed environment. In the future, we will further investigate the relationship between the ensemble model accuracy and the number of features selected and the relationship between the ensemble model accuracy and the number of models. We will also try other methods to select different feature subsets based on the random subspace method. Besides, we will also investigate the new methods to evaluate the weight of different models and to use weighted voting method to predict classification result.

References

1. Lin, C., Weng, R., Keerthi, S.: Trust region newton methods for large-scale logistic regression. In: Proceedings of the 24th International Conference on Machine Learning, pp. 561–568. ACM (2007)
2. Minka, T.: A comparison of numerical optimizers for logistic regression (2003) (unpublished draft)
3. Komarek, P., Moore, A.: Making logistic regression a core data mining tool with tr-irls. In: International Conference on Data Mining, pp. 685–688 (2005)
4. Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), vol. 20, p. 856 (2003)
5. Prinzie, A., Van den Poel, D.: Random forests for multiclass classification: Random multinomial logit. Expert Systems with Applications 34(3), 1721–1732 (2008)
6. Dean, J., Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. Communications of the ACM 51(1), 107–113 (2008)
7. White, T.: Hadoop: The definitive guide. Yahoo Press (2010)
8. Venner, J.: Pro Hadoop. Springer (2009)
9. Lam, C., Warren, J.: Hadoop in action (2010)
10. Jennrich, R., Sampson, P.: Newton-raphson and related algorithms for maximum likelihood variance component estimation. Technometrics, 11–17 (1976)
11. Komarek, P.: Logistic regression for data mining and high-dimensional classification. Doctoral Thesis (2004)
12. Ho, T.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
13. Dietterich, T.: Ensemble methods in machine learning. Multiple Classifier Systems, 1–15 (2000)
14. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
15. Witten, I., Frank, E., Trigg, L., Hall, M., Holmes, G., Cunningham, S.: Weka: Practical machine learning tools and techniques with java implementations. In: ICONIP/ANZIIS/ANNES, vol. 99, pp. 192–196. Citeseer (1999)

Verifying Location-Based Services
with Declassification Enforcement⋆

Cong Sun, Sheng Gao, and Jianfeng Ma

School of Computer Science and Technology, Xidian University, China
suncong@xidian.edu.cn

Abstract. Location privacy has been considered to be an important issue along with the advances of location technologies and the pervasive use of location based services. Although a variety of location privacy techniques have been developed, language-based techniques have rarely been used on privacy enforcement of location-based services. In this work, we propose a verification framework to enforce the privacy preservation of locations. The enforcement leverages reachability analysis of pushdown system to ensure that the service-specific data aggregation functions comply with the privacy property. The approach supports inter-procedural analysis and is more precise than existing work.

1   Introduction

Location-based services (LBS) are a category of services that integrate the location information of mobile nodes with other information to provide easily accessible added value to mobile users. LBS usually require users' current location information to answer their location-based queries. Along with the widely use of LBS, user that employs LBS frequently faces potential privacy problem. The privacy of user is threatened as LBS require the user to release her private location information to them and these data allow for profiling the user's movement, health condition, or affiliation. Therefore Developing enforcement mechanisms of privacy requirements to preserve the private location information for mobile users becomes an emergency. There have been several paradigms of architectures for privacy-preserving LBS [3,12] where the trade-offs between anonymity of private location information and the quality of service are discussed. In the *trusted third party architecture*, a static analyzer can be developed on the trusted server to ensure the data aggregation function of private location information can meet with certain privacy requirement. By verifying the possibly malicious aggregation function code from the untrusted LBS provider, this approach can provide a service-specific privacy without loss of quality of service.

In this work we propose a verification framework for the trusted server to enforce privacy property that prevent unintentional leak of users' private location

information. The typical application scenario is given in Fig. 1. It involves three kinds of parties: 1) the trusted server maintaining all the location information of mobile nodes, 2) the location information provider on each mobile node which sends private location information to the trusted servers, 3) the LBS which defines service-specific data aggregation function with a piece of code, sends the code to the trusted servers for computation and receives the results of computation from the trusted servers. The aggregation function developers use explicit outputs to claim the location information the LBS requires. But without careful examination, the malicious or cursorily developed aggregation function may leak the private location information in a unintentional way. Therefore a verification framework is needed by the trusted servers to ensure this unintentional leakage cannot happen.

The trusted servers first use symbolic pushdown system to model the aggregation functions, then use reachability analysis to decide whether the declassified results of computation meet with the privacy property specified in Section 2.2. If the aggregation function satisfies the pri-

Fig. 1. Application Scenario

vacy property, the trusted server feeds the function with the location information and the data from LBS, and sends the result of computation back to the LBS. We transform the pushdown model of aggregation function using a form of self-composition [1] complemented with a set of auxiliary pushdown rules in order to reduce the declassification-based privacy property to a safety property on the model after transformation. The reachability analysis is then performed on the derived model. A store-match pattern is inherited from our previous work to reduce the state space and to constrain the pre-condition of privacy property.

Numerous studies have focused on privacy issues in LBS. The pseudonym-based approaches, which prevent any user to communicate with the services [2,5], may greatly influence the QoS. An alternative category of approaches is obfuscation [4,8]. These work mostly focus on the anonymity of locations. On the other hand, our approach mainly focuses on how to validate a privacy-preserved usage, i.e. data aggregation, of users' location information. The most closely related work is that of Ravi et al [6]. They specify the privacy requirement with the *non-inference* property and use a global data-flow analysis to solve whether any explicit flow of location information to output variables exists. Compared with our work, their approach imprecisely reports more false-negatives, and cannot support inter-procedural analysis.

2   Approach

2.1   Reachability Analysis of Pushdown System

We use symbolic pushdown system [9] as the model of aggregation function code. A pushdown system is a stack-based state transition system whose stack
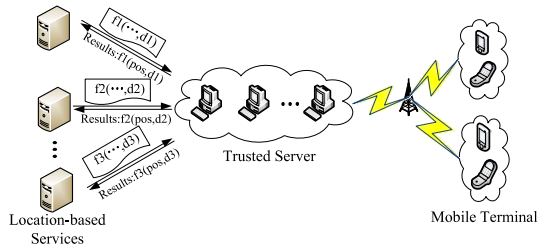
Table 1. Model Construction

c	$\Phi(c, \gamma_j, \gamma_k)$
skip	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; rt(\mu)\}$
$x := e$	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; (x' = e) \wedge rt(\mu \setminus \{x\})\}$
if e then c_1 else c_2	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_t\rangle \; rt(\mu) \wedge e\} \cup \Phi(c_1, \gamma_t, \gamma_k) \cup$
	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_f\rangle \; rt(\mu) \wedge \neg e\} \cup \Phi(c_2, \gamma_f, \gamma_k)$
while e do c'	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_t\rangle \; rt(\mu) \wedge e\} \cup \Phi(c', \gamma_t, \gamma_j) \cup \{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; rt(\mu) \wedge \neg e\}$
$c_1; c_2$	$\Phi(c_1, \gamma_j, \gamma_{\mathrm{mid}}) \cup \Phi(c_2, \gamma_{\mathrm{mid}}, \gamma_k)$
$x := f(y)$	$\{\langle\gamma_j\rangle \hookrightarrow \langle f_{\mathrm{entry}}\gamma_t\rangle \; (z' = y) \wedge rt(\mu_{\mathrm{glb}}) \wedge rt_2(\mu_{\mathrm{loc}})\} \cup$
	$\{\langle f_{\mathrm{exit}}\rangle \hookrightarrow \langle\epsilon\rangle \; (iret' = \ldots) \wedge rt(\mu_{\mathrm{glb}} \setminus \{iret\})\} \cup$
	$\{\langle\gamma_t\rangle \hookrightarrow \langle\gamma_k\rangle \; (x' = iret) \wedge rt(\mu \setminus \{x\})\}, \quad (z \in \mu_{\mathrm{loc}_f})$
$output(x)$	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; (O'[q] = x) \wedge (q' = q + 1) \wedge rt(\mu \setminus \{O[q], q\})\}$

contained in each state is of unbounded length. A symbolic pushdown system is a compact representation of pushdown system encoding the variables and computations symbolically.

Definition 1 (Symbolic Pushdown System). *Symbolic Pushdown System is a triple $\mathcal{P} = (\mathcal{G}, \Gamma \times \mathcal{L}, \Delta)$. $\mathcal{G}$ and $\mathcal{L}$ are respectively the domain of global variables and local variables. Γ is the stack alphabet. Δ is the set of symbolic pushdown rules $\{\langle\gamma\rangle \hookrightarrow \langle\gamma_1 \cdots \gamma_n\rangle(\mathcal{R}) \mid \gamma, \gamma_1, \cdots, \gamma_n \in \Gamma \wedge \mathcal{R} \subseteq (\mathcal{G} \times \mathcal{L}) \times (\mathcal{G} \times \mathcal{L}^n) \wedge n \leq 2\}$.*

The relation $\mathcal{R}$ specifies the environment transformers that direct a single step of symbolic computation according to the pushdown rule. The stack symbols denote the flow graph nodes of program. The model construction function Φ is given in Table 1. It adds constraints to regulate each $\mathcal{R}$ of pushdown rule. The constraint is expressed with logical operation on abstract variables. $\mu = \mu_{\mathrm{glb}} \cup \mu_{\mathrm{loc}_f}$ is a memory store mapping both the global variables and the local variables of current procedure f to values. For the abstraction of output command, we suppose the global linear list O and the index q for next output element to have $\{O, q\} \subseteq dom(\mu_{\mathrm{glb}})$. $iret$ is a global variable temporarily storing the returned value of f. rt means retainment on value of global variables and on value of local variables of the procedure locating the pushdown rule. rt_2 for a rule $\langle\gamma_j\rangle \hookrightarrow \langle f_{\mathrm{entry}}\gamma_k\rangle$ denotes retainment on value of local variables of the caller of procedure f. The back-end algorithm we used for reachability analysis is the symbolic form of saturation algorithm $post^*$ adapted from [9, Sec.3.3.3]. The reachability analysis of certain flow graph node γ actually finds whether the $\mathcal{P}$-automaton $\mathcal{A}_{post^*}$ can accept a configuration whose top stack symbol is γ.

2.2 Specification of Privacy

From a perspective of trusted third party, the LBS should explicitly specify what they want to learn from the private location information, e.g., through the outputs and explicit declassification primitives in their code [6]. The trusted server ensures the explicit outputs of a single run of aggregation function will not release the private location itself. In this work, we use the end-to-end enforcement

of declassification requirement to ensure that the intentional release of information cannot leak any confidential information other than what the LBS claim to obtain through the explicit outputs.

For any aggregation function f, the set of parameters w.r.t. LBS data is $L = \{\ell_1, \ldots, \ell_m\}$ and the set of parameters w.r.t. private location information is $H = \{h_1, \ldots, h_p\}$. Then we have $L \cup H \subseteq dom(\mu_{\mathrm{loc}_f})$. For the test case in [6, Fig.2], i.e. `DistCoord` in this work, $H = \{$x1,x2,y1,y2$\}$ and $L = \{$k$\}$. We suppose that $\mu(e)$ is the evaluation of expression e under μ, and $(\mu, c) \Downarrow \mu'$ means that execution of c under μ derives μ'. We define the set-related indistinguishability $\mu =_L \mu'$ means $\forall \ell \in L.\mu(\ell) = \mu'(\ell)$, and $\mu =_{\{O,q\}} \mu'$ means $(q = q') \wedge (\forall 0 \leq i < q.O[i] = O'[i])$. The declassification-based privacy property is specified as follow.

Definition 2 (Privacy Preservation). *Data aggregation function f preserves privacy, iff $\forall \mu_1, \mu_2 : (\mu_1 =_L \mu_2 \wedge \forall 0 \leq i < n.\mu_1(e_i) = \mu_2(e_i) \wedge (\mu_1, f_{body}) \Downarrow \mu_1' \wedge (\mu_2, f_{body}) \Downarrow \mu_2') \Rightarrow (\mu_1' =_{\{O,q\}} \mu_2')$, where f_{body} is the command w.r.t. the body of f, and $e_i(0 \leq i < n)$ is (1) the expression which contains some $h_k \in H$ and is used to obtain the value of some explicit output variable, or (2) the expression explicitly released with declassification primitives.*

From the definition, we know the value of expression $e_i(0 \leq i < n)$ are what the LBS claimed to get, and we treat these expressions to be declassifiable. For `DistCoord`, the declassifiable expressions are $\{$(x2-x1)^2, (y2-y1)^2, (x1+x2)/2, (y1+y2)/2$\}$. The declassified values of these expressions are actually decided by the private location information and the LBS data which are passed to the aggregation function by parameters. The treatment of public input, i.e. variables in L, is different from that for delimited release [7]. We do not judge the indistinguishability on L at final states because even the LBS data passed by L can only be sent back to LBS through explicit output. The privacy property indicates that if the variation of confidential information (typically the private location information carried by parameters in H) cannot cause variation of initial value of e_i, then the variation of confidential information will not be detected through the public output. On the other hand, from the violation of property, i.e. variation of public outputs, we can learn that the leakage of confidential information is not caused by the declassifiable e_i.

2.3   Declassification Enforcement

We propose a reachability analysis to enforce the declassification-based privacy property. We develop our enforcement based on the model transformation algorithm in [11, Sec.IV]. With the transformation, the declassification-based privacy property can be reduced to a safety property that is enforceable using reachability analysis. The model is transformed with a variant of *compact self-composition* [10] facilitated with a set of auxiliary pushdown rules to perform the interleaving assignments and to construct the illegal-flow states. Also, the initial equivalences on declassifiable expressions are considered in these auxiliary pushdown rules.

Table 2. Compact Self-Composition

r	$\mathcal{SC}(r)$
$\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; \mathcal{R}$	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; \mathcal{R} \wedge rt(\xi(\mu \setminus \{O,q\})),$ $\langle\xi(\gamma_j)\rangle \hookrightarrow \langle\xi(\gamma_k)\rangle \; \mathcal{R}^{\xi(x)}_{x\in L\cup H} \wedge rt(\mu \setminus \{O,q\})\}$
$\langle\gamma_j\rangle \hookrightarrow_o \langle\gamma_k\rangle \; \mathcal{R}$	$\{\langle\gamma_j\rangle \hookrightarrow \langle\gamma_k\rangle \; \mathcal{R} \wedge rt(\xi(\mu \setminus \{O,q\})),$ $\langle\xi(\gamma_j)\rangle \hookrightarrow \langle\xi(\gamma_k)\rangle \; (O[q]=x) \wedge (q'=q+1)\wedge$ $rt(\mu \setminus \{q\}, \xi(\mu \setminus \{O,q\})),$ $\langle\xi(\gamma_j)\rangle \hookrightarrow \langle error\rangle \; (O[q]\neq x)\wedge rt(\mu, \xi(\mu \setminus \{O,q\}))\}$
$\langle\gamma_j\rangle \hookrightarrow \langle g_{\text{entry}}\gamma_k\rangle \; \mathcal{R}$	$\{\langle\gamma_j\rangle \hookrightarrow \langle g_{\text{entry}}\gamma_k\rangle \; \mathcal{R}\wedge$ $rt(\xi(\mu_{\text{glb}} \setminus \{O,q\})) \wedge rt_2(\xi(L\cup H)),$ $\langle\xi(\gamma_j)\rangle \hookrightarrow \langle\xi(g_{\text{entry}})\xi(\gamma_k)\rangle \; \mathcal{R}^{\xi(x)}_{x\in L\cup H}\wedge$ $rt(\xi(\mu_{\text{glb}} \setminus \{O,q\})) \wedge rt_2(L\cup H)\}$
$\langle\gamma_j\rangle \hookrightarrow \langle\epsilon\rangle \; \mathcal{R},$ $(\gamma_j \neq f_{\text{exit}})$	$\{\langle\gamma_j\rangle \hookrightarrow \langle\epsilon\rangle \; \mathcal{R} \wedge rt(\xi(\mu_{\text{glb}} \setminus \{O,q,iret\})),$ $\langle\xi(\gamma_j)\rangle \hookrightarrow \langle\epsilon\rangle \; \mathcal{R} \wedge rt(\xi(\mu_{\text{glb}} \setminus \{O,q,iret\}))\}$
$\langle f_{\text{exit}}\rangle \hookrightarrow \langle\epsilon\rangle \; \mathcal{R},$	$\{\langle\xi(f_{\text{exit}})\rangle \hookrightarrow \langle\xi(f_{\text{exit}})\rangle \; \mathcal{R}^{\xi(x)}_{x\in\mu\setminus\{O,q\}} \wedge rt(\mu \setminus \{O,q\})\}$

Table 3. Refinement of Auxiliary Pushdown Rules

PDS Rule	Refined PDS Rules
$\langle start\rangle \hookrightarrow \langle f_{\text{entry}}\rangle \; \mathcal{R}_1$	$\{\langle start\rangle \hookrightarrow \langle\gamma_{\text{mid}_1}\rangle \; \mathcal{R}_1,$ $\langle\gamma_{\text{mid}_1}\rangle \hookrightarrow \langle f_{\text{entry}}\rangle \; (\bigwedge_{0\leq i<n} \mathcal{D}'[\rho(e_i)]=e_i)\wedge$ $rt(\mu, \xi(\mu \setminus \{O,q\}))\}$
$\langle f_{\text{exit}}\rangle \hookrightarrow \langle\xi(f_{\text{entry}})\rangle \; \mathcal{R}_2$	$\{\langle f_{\text{exit}}\rangle \hookrightarrow \langle\gamma_{\text{mid}_2}\rangle \; \mathcal{R}_2,$ $\langle\gamma_{\text{mid}_2}\rangle \hookrightarrow \langle\xi(f_{\text{entry}})\rangle \; (\bigwedge_{0\leq i<n} \mathcal{D}[\rho(e_i)]=\xi(e_i))\wedge$ $rt(\mu, \xi(\mu \setminus \{O,q\})),$ $\langle\gamma_{\text{mid}_2}\rangle \hookrightarrow \langle idle\rangle \; (\bigvee_{0<i<n} \mathcal{D}[\rho(e_i)]\neq\xi(e_i))$

The compact self-composition $\mathcal{SC}$ used here is given in Table 2. ξ is the rename function on stack symbols to generate new flow graph nodes and on variables to generate companion variables for the model of second run. $\mathcal{R}^{\xi(x)}_{x\in V}$ is a relation derived by substituting each variable in V with the renamed companion variable. We annotate the abstraction of explicit output with $\hookrightarrow_o$, and then construct the illegal-flow state *error* within the compact self-composition by comparing the output x of the second run with the corresponding output stored in the first run to judge if they are not equal. $\mathcal{SC}$ is applied on each pushdown rule r in $\Phi(f_{\text{body}}, f_{\text{entry}}, f_{\text{exit}})$ to derive a set of pushdown rules $\bigcup_{r\in\Phi(f_{\text{body}},f_{\text{entry}},f_{\text{exit}})} \mathcal{SC}(r)$. The result of compact self-composition still needs two auxiliary pushdown rules:

1. $\langle start\rangle \hookrightarrow \langle f_{\text{entry}}\rangle \; \mathcal{R}_1$: This pushdown rule is used to perform initial interleaving assignments and to set the initial index q to 0;
2. $\langle f_{\text{exit}}\rangle \hookrightarrow \langle\xi(f_{\text{entry}})\rangle \; \mathcal{R}_2$: This pushdown rule is used to reset q for the reuse of output channel in the second run.

The initial interleaving assignments can be specified with

$$\mathcal{R}_1 \equiv (\forall \ell \in L.\ell' = \xi(\ell)) \wedge (q' = 0) \wedge rt(\mu \setminus \{L, q\}, \xi(\mu \setminus \{O, q\}))$$

and the index q can be reset with

$$\mathcal{R}_2 \equiv (q' = 0) \wedge rt(\mu \setminus \{q\}, \xi(\mu \setminus \{O, q\}))$$

Both pushdown rules are also involved in modeling the initial equivalence of declassifiable expressions. More specifically, the modeling of the initial $\mu_1(e_i) = \mu_2(e_i)$ leverages the store-match pattern [11] on auxiliary list $\mathcal{D}$. We define a function $\rho : \{e_i \mid 0 \le i < n\} \mapsto \{0..n-1\}$ to map each declassifiable expression to the index of $\mathcal{D}$. Then we refine the above two auxiliary pushdown rules to add the store-match operations. The results are given in Table 3. For the case `DistCoord`, these refined pushdown rules are used to model

```
int f(...){
    k=$k; q=0;
    D[0]=(x2-x1)^2; D[1]=(y2-y1)^2; D[2]=(x1+x2)/2; D[3]=(y1+y2)/2;
    //model of the first run
    q=0;
    if(D[0]!=($x2-$x1)^2 || D[1]!=($y2-$y1)^2 ||
        D[2]!=($x1+$x2)/2 || D[3]!=($y1+$y2)/2) goto idle;
    //model of the second run
}
```

Reaching the state *idle* implies violation of some $\mu_1(e_i) = \mu_2(e_i)$, and whenever *idle* is reached, the illegal-flow state *error* becomes unreachable. Therefore the reachability of *idle* rules out the trace irrelevant to the judgement of postcondition of privacy preservation.

3   Evaluations

We modified our previous implementation [11] to support the privacy property verification. The aggregation functions are modeled with Remopla. The test cases are selected from related work to evaluate the preciseness of our enforcement compared with the global data-flow analysis [6], see Table 4. The length of DB we used for `PasswordChecker` is 100. `Density` and `CoordAvg` are manually implemented aggregation function. `Density` counts the number of nodes in a specific rectangle area, and `CoordAvg` calls `Avg` to derive the average coordinates of a set of nodes. The interfaces are like

```
int Density(int X[N],int Y[N],int up,int down,int left,int right);
int CoordAvg(int X[N],int Y[N]);
```

In Table 4, Non-Inf means the satisfaction of non-inference property, and GDF is the result of global data-flow analysis. RA is the result of our reachability analysis. *len* and *bit* are two factors of model that may have impact on the efficiency of our verification. *len* denotes the length of output channel, and *bit* is the number of bits for integer variables and elements in channel. T_{bit}^{len} is the time for reachability analysis of model with parameter *len* and *bit*. len_{min} is

Table 4. Precision (2.83GHZ×4 Intel CPU, 4GB RAM, Linux 2.6.38-8-generic)

Case	From	Non-Inf[6]	GDF	RA	len_{min}	bit_{min}	$T^{len_{min}}_{bit_{min}}$(s)	T^5_2(s)
DistCoord	Fig.2,[6]	✓	✓	✓	3	1	≤0.01	0.01
Avg(n=10)	Sec.3.3,[7]	✓	✓	✓	1	1	0.01	61.95
AvgAttack(n=10)	Sec.3.3,[7]	×	×	×	1	1	0.02	62.58
Wallet	Sec.3.3,[7]	✓	×	✓	1	1	≤0.01	≤0.01
WalletAttack	Sec.3.3,[7]	×	×	×	1	2	≤0.01	0.01
ParityAttack	Sec.3.4,[7]	×	×	×	1	2	≤0.01	≤0.01
Parity	Sec.3.4,[7]	✓	×	✓	1	1	≤0.01	≤0.01
ParitySec	Sec.4,[7]	✓	×	✓	1	1	≤0.01	≤0.01
PasswordChecker	Sec.5,[7]	✓	✓	✓	1	1	0.14	4.26
PasswordUpdate	Sec.5,[7]	✓	✓	✓	1	1	≤0.01	0.02
Density(n=100)	–		✓	✓	✓	1	1	0.06	37.66
CoordAvg(n=100)	–		✓	✓	✓	2	1	0.05	16.08

the minimum length of output channel that is sufficient to identify the possible inequality of output sequences, i.e. the violation of post-condition $\mu'_1 =_{\{O,q\}} \mu'_2$ of privacy property. bit_{min} is the minimum bit that is required to avoid the false-positive. With a large enough bit, the false-positive is always avoidable therefore it will not influence the final preciseness of our approach. Meanwhile, our approach is more precise than the global data-flow analysis to avoid the false-negatives appeared in `Wallet`, `Parity`, and `ParitySec`. The false-negative is mainly cause by implicit information flow that can be avoid by our value-dependent approach. Although the global data-flow analysis is flow-sensitive and correctly judges the program `l=h1+h2;l=0` to be secure, the following program

```
if(h) {l=1} else {l=h+1}; output(l);
```

is mistakenly judged insecure by the global data-flow analysis. On the contrary, our value-dependent analysis can ensure it preserves privacy of the initial `h`.

From the comparison of $T^{len_{min}}_{bit_{min}}$ and T^5_2 we know the increase on value of either len or bit may cause the increase on the verification time. We use further experiment to evaluate the impact of len and bit on the efficiency of our verification. The impact is illustrated with the experimental results given in Fig. 2. The left side figures show the variation of verification time w.r.t. different length of output channel. In these experiments $bit = 2$ and the length of channel is set from 5 to 50 with common difference 5. We can see that the increase on length of output channel will not necessarily increase the verification time. The right side figure shows the variation of verification time when bit is increased and len is set to 5. We can see the verification time increases exponentially as the value of bit increases. Larger size of array leads to a faster increase on verification time. For example, `Avg`, `AvgAttack`, `Density`, `CoordAvg` and `PasswordChecker` cause a time limitation exceeded when $bit = 3$, while for `Parity`, `ParityAttack`, `ParitySec` and `WalletAttack`, we meet the TLE at $bit = 9$.
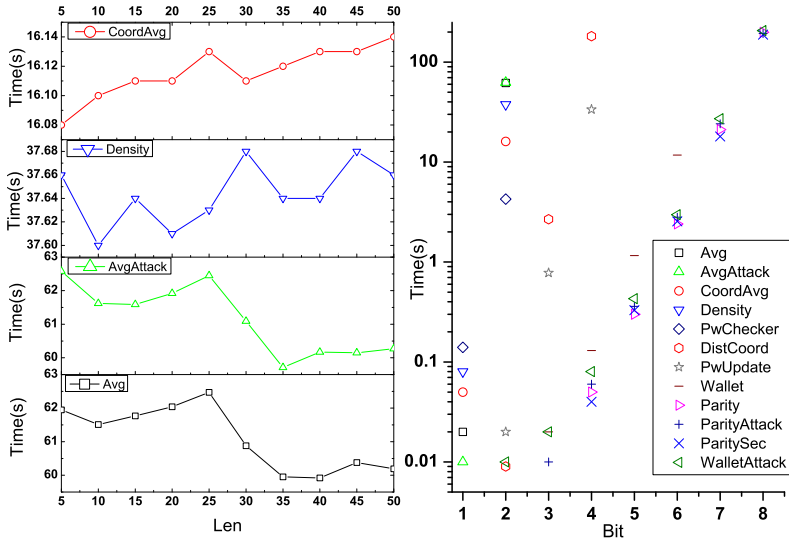
Fig. 2. Impact of *len* and *bit* on Efficiency of Verifications

References

1. Barthe, G., D'Argenio, P.R., Rezk, T.: Secure information flow by self-composition. In: CSFW, pp. 100–114. IEEE Computer Society (2004)
2. Beresford, A., Stajano, F.: Location privacy in pervasive computing. IEEE Pervasive Computing 2(1), 46–55 (2003)
3. Chow, C.Y., Mokbel, M.F.: Privacy in location-based services: a system architecture perspective. SIGSPATIAL Special 1(2), 23–27 (2009)
4. Hong, J.I., Landay, J.A.: An architecture for privacy-sensitive ubiquitous computing. In: MobiSys 2004, pp. 177–189. ACM, New York (2004)
5. Jiang, T., Wang, H.J., Hu, Y.C.: Preserving location privacy in wireless lans. In: MobiSys 2007, pp. 246–257. ACM, New York (2007)
6. Ravi, N., Gruteser, M., Iftode, L.: Non-inference: An information flow control model for location-based services. In: 3rd International Conference on Mobile and Ubiquitous Systems Workshops, pp. 1–10 (2006)
7. Sabelfeld, A., Myers, A.: A Model for Delimited Information Release. In: Futatsugi, K., Mizoguchi, F., Yonezaki, N. (eds.) ISSS 2003. LNCS, vol. 3233, pp. 174–191. Springer, Heidelberg (2004)
8. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Tech. Rep. SRI-CSL-98-04, SRI International (1998)
9. Schwoon, S.: Model Checking Pushdown Systems. Ph.D. thesis, Technical University of Munich, Munich, Germany (2002)
10. Sun, C., Tang, L., Chen, Z.: Secure information flow by model checking pushdown system. In: UIC-ATC 2009, pp. 586–591. IEEE Computer Society (2009)
11. Sun, C., Tang, L., Chen, Z.: A new enforcement on declassification with reachability analysis. In: INFOCOM 2011 Workshops, pp. 1024–1029. IEEE (2011)
12. Yiu, M.L., Jensen, C.S., Møller, J., Lu, H.: Design and analysis of a ranking approach to private location-based services. ACM Trans. Database Syst. 36, 1–42 (2011)

Single-Hop Friends Recommendation
and Verification Based Incentive for BitTorrent

Yan Pang and Zongming Guo

Institute of Computer Science and Technology
Peking University, China
{yan.pang,guozongming}@pku.edu.cn

Abstract. Free riding, the behavior of attempting to benefit resources contributed by others while sharing their own values as minimum as possible, is one of the key problems in many P2P systems. In this paper, we propose an incentive mechanism based on single-hop friends' recommendation and verification to improve the performance of BitTorrent. Peers figure out non-free-riders from their personal experience and friends' recommendation and verification, and optimistic unchoke one peer which is randomly selected from non-free-riders during its optimistic unchoke period. With the help of the modified seed unchoke algorithm, the proposed incentive mechanism prevents the free riding behavior and the collusion of malicious peers effectively.

Keywords: free riding, incentive, single-hop, recommendation.

1   Introduction

Ipoque Internet study for years 2008-2009 shows that P2P generates the most traffic ranging from 43% to 70% in different regions. The study also finds that BitTorrent is still the dominant protocol on the Internet, and it accounted for approximately 27% to 55% of all Internet traffic depending on the geographical location [1]. However, BitTorrent is prone to free riding or strategic attacks. Several modified BitTorrent clients [2,3] have been developed which exploit different strategies to achieve better performance at the expense of users running unmodified BitTorrent. BitThief is a free riding client whose behavior is described in [2]. BitThief exploits the altruism present in the swarm by procuring a large peers list in the swarm. It then attempts to establish connections with seeders and procures free content. While interacting with other peers, BitThief adopts a policy of uploading minimum content to get the maximum download. One of the very important exploits BitThief used is to pretend to be a newcomer (advertises having no piece) to get more optimistic unchoke slot from leechers. It also pretends being a great contributor in sharing communities by announcing bogus information to cheat the tracker and get a higher sharing ratio without ever uploading a single bit. [2] demonstrates that BitThief can perform better than regular clients with scenarios where majority of the client run the regular BitTorrent. BitTyrant [3] is another modified client which strategically tries to

maximize its download rates by dynamically adapting and shaping the upload bandwidth allocated to its neighbors.

Tit-for-Tat incentive mechanism is employed in BitTorrent, but [4] suggests that the Tit-for-Tat is not efficient enough in deterring unfairness and relates this inadequacy to the heterogeneity of peers' bandwidths. Moreover, [5] argues that BitTorrent lacks fairness: It does not punish free riders effectively neither it does reward users who contribute properly. [6] presents an experimental study on the behavior of BitTorrent. The results show that a free rider can perform better than a compliant peer. [7] points out that the optimistic unchoke mechanism in BitTorrent opens up the client to losing up to 20% of its bandwidth to free riders who simply need to wait for other peers to "optimistically" unchoke them.

In this paper, we propose a single-hop friends' recommendation and verification based incentive mechanism for BitTorrent. The proposed mechanism regards the recommendation and verification as services likewise, and motivates peers to share file with non-free-riders, and recommend and verify non-free-riders. All the non-free-riders which checked by a peer itself, and recommended and verified by friends form an optimistic unchoke candidate set. The modified optimistic unchoke mechanism randomly selects a peer from this non-free-rider candidate set. With the auxiliary of the modified seed unchoke algorithm, our incentive mechanism prevents the free riders and the collusion of malicious peers effectively. Our work has much in common with recent work on the design of reputation systems for various P2P applications. In [8], Lian *et al.* proposed multi-level tit-for-tat incentives as a hybrid between private and shared history schemes. More recently, Piatek *et al.* proposed a one-hop reputation system [9], in which reputation propagation is limited at most one intermediary, and peers that are not interested in a current available content perform work in exchange for the assurance of future payback. However, these reputation systems require significant communication overhead to maintain the global history. Some of them compute the reputation value for all participants relying on a small number of trusted peers. Furthermore, even if the computed reputation value accurately represents peer contribution behavior, there is no guarantee that each peer expresses the same behavior to different peers with different attributes. Also, most of them are open to collusion especially where majority of the peers are malicious and can forge one another reputation.

We describe the the novel incentive mechanism based on single-hop friends' recommendation and verification in Section 2. Section 3 presents the simulation results. Finally, we describe future work in Section 4.

2   Single-Hop Friends' Recommendation and Verification Based Incentives

In this section, we describe a new, single-hop friends' recommendation and verification based incentive mechanism to enable long-term and indirect reciprocation. The main goal of this novel incentive mechanism is to foster persistent contribution incentives by recognizing and rewarding contributors across the whole

swarm and over time but not only limited to a specific peer and in a short time. The key idea of our scheme is to find out the non-free-riders by the peer's it-self or by the single-hop friends' recommendation and verification, and maintain a non-free-rider candidate set on each peer from which the optimistic unchoke mechanism randomly chooses the peer. The non-free-rider here is judged from a global view throughout a whole swarm but not from a local view of a specific peer. The single hop restricts the amount of indirection between contributing and reciprocating peers to at most one level of intermediaries. This restriction limits the propagation of information, promoting scalability, and allows for local reasoning about the trustworthiness of intermediaries.

2.1 Who Is the Free Rider or Who Is the Non-free-rider

Because BitTorrent is a large and dynamic system, a peer often interacts with strangers with no prior history or reputation to it. It is very important to think about how one deal with strangers. Optimistic unchoke strategy that always cooperates with strangers may encourage newcomers to join the system, but it can be easily exploited by free riders and strategic selfish peers such as *BitThief*s who are free riders but always announce themselves as newcomers of the system. On the other hand, always defecting against strangers is robust against free riders and strategic selfish peers, but it discourages newcomers to join the system. If a peer can figure out who are the free rider and who are not the free-rider when it meets other peers for the first time, it could properly decide whom it should provide service to, and the bandwidth wasted on free riders could be saved.

Table 1 and Table 2 list each role of a peer from the global view and local view separately. A stranger of peer A might be a newcomer of the system or a peer which has already joined the system and exchanged file pieces with other peers but has not interacted with peer A.

Table 1. Roles Definition from Global

Role	Definition
Newcomer of the system	peer who joins the BT system for the first time without any file piece
Free rider of the system	peer who obtains resource from but contributes nothing to the system
Contributor of the system	peer who has send the file pieces to other peers in the system

Table 2. Roles Definition from Local

Role	Definition
Stranger of a specific peer	peer who interacts with the specific peer for the first time
Friend of a specific peer	peer who sends file pieces to the specific peer
Recommender of a specific peer	peer who recommends the specific peer to other peers and is verified by other peers
Friend of friend	peer who is recommended by a friend of the specific peer

Peer A could not identify peer B as a free rider of the system only from peer B's behavior of returning nothing to A after obtained resource from A. If peer B cooperate with other peers, it is not a free rider but a contributor of the system. A friend of a peer must be a contributor of the system.
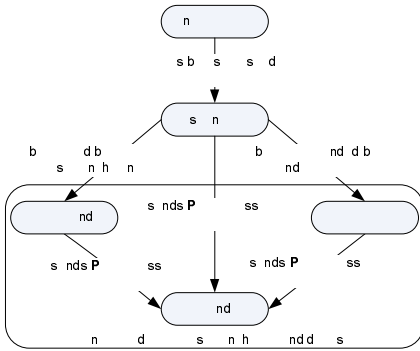
Fig. 1. Roles Transition Diagram
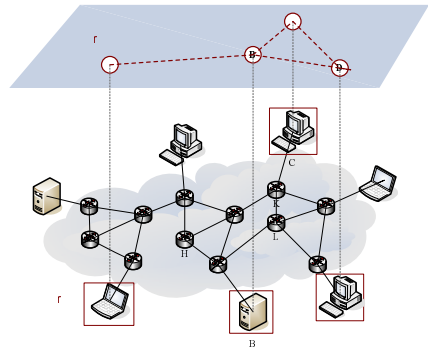
Fig. 2. An Example of P2P Overlay

A peer could not figure out what a stranger is, a newcomer, a free rider, or a strategic selfish peer? However, a peer could find out some non-free-riders of the system with its friends' help. We could infer that if a peer who has contributed to some peers is not a free rider of the system, it will contribute to other peers if it has the ability in the future. Thus, we improve the optimistic unchoke mechanism by unchoking non-free-rider of the system randomly. The free riders and the strategic selfish peers could not utilize the optimistic unchoke mechanism to obtain the data without contributing anything.

2.2   Single-Hop Friends' Recommendation and Verification

Most of the cautious individuals may only want to rely on their own personal experience and use only local information when determining whether to transact with a given peer. To increase the information sources which can assist the decision, the individuals can collect the opinions from users, for example their friends, whom they have a priori trust relationships with.

In the original BitTorrent protocol, the optimistic unchoke strategy chooses a peer to unchoke randomly regardless of whether it is a free rider. Some free riders and strategic selfish peers exploit this shortcoming to obtain the file pieces without contributing anything. We propose an improved optimistic unchoke mechanism that every peer sets up a non-free-rider candidate set which collects non-free-riders of the system. Recommendation is regarded as a service like bandwidth. We use the friends' recommendation and verification as the assistant information. If peer A recommends peer C to peer B, we say that peer A provides service to peer B even if it does not provide any file piece to peer B.

When peer A unchokes and sends some data to peer B, peer B will check the accuracy of the data. After sending a specific amount of accurate data, peer A is enabled to recommend one of its friends, peer C, to peer B as a non-free-rider of the system. When peer B optimistically unchokes peer C, it will verify that peer A is peer C's recommender.

We collect the peer's friends, its friends of friends, and its recommenders to form a non-free-rider optimistic unchoke candidate set. Figure 1 shows the process of a peer's role transition and the elements of the non-free-riders optimistic unchoke candidate set. The member of optimistic unchoke candidate set can not be duplicated. A peer which has been in the set could not be added again when it was recommended by other peers. In this set, all the peers are non-free-riders of the system. At the beginning of optimistic unchoking period, the source peer randomly selects a peer from the optimistic unchoke candidate set and unchokes it. Whether all these candidates will provide upload service to the peer will be checked by the regular unchoke mechanism.

Both the recommendation and the verification of a peer can be accepted only after it provided data to the destination peer, and the data provided should be accurate. That means if a peer wants to be a recommender or a verifier, it must provide another peer some accurate data. This can prevent the strategic selfish peer from disguising itself as a friend and recommending or verifying other free riders as non-free-riders. The more the total accurate data a peer provided, the more its friends are recommended, then the chance of becoming a recommender will increase, and thus the more optimistic unchoke opportunities it could obtain. This can incentive peers to provide more services to others. We can simply use the Fast Extension mechanism to limit the free data obtained by strategic selfish peers who always announce themselves as newcomers. Algorithm 1 and 2 show the processes of a peer becoming a friend, a recommender, and a recommendee.

Algorithm 1. The role transition of peer p_i's neighbors

---

1 **if** p_j *send* $data_{ji}$ *to* p_i **then**
2   friend.push(j, $data_{ji}$);
3   **if** $(data_{ji} \geq friend_{thresh}) \wedge (data_{ji} \leq recommender_{thresh})$ **then**
4   nonfreerider.push(j);
5   **end**
6   **if** $(data_{ji} \geq recommender_{thresh})$ **then**
7         $recommendee \leftarrow$ getPiggyback(PIECE);
8   nonfreerider.push($recommendee, j$);
9   **end**
10 **end**

Algorithm 2. The peer p_i's optimistic unchoke algorithm in enhanced BitTorrent

---

1 int $i = 0$;
2 i=Rand()%$numOfNonfreerider$;
3 $opt\_unchoke \leftarrow$ nonfreerider[i].getPeer();
4 $recommender \leftarrow$ nonfreerider[i].getRecommender();
5 messagePiggyback(OPTUNCHOKE, $recommender$);
6 sendMessage($opt\_unchoke$, OPTUNCHOKE);

We do not deploy an accurate reputation system but only a simple recommendation strategy because the metric for reputation value is local, context-aware and behavior sensitive. In Client/Server mode, the reputation is more accurate because the architecture is stable, the server is on duty most of the time and the bandwidth between servers and clients vibrates little. While in P2P systems, the environment is dynamic. Peers join and leave the system without schedule in advance, and the bandwidth between peers fluctuates heavily. At the same time, some peers exhibit asymmetric interest. So the calculated reputation value in P2P systems could not represent the real situation. The inconsistency between the overlay network and the underlay network also affects the accuracy of the reputation value. In the overlay network in figure 2, peer B is a friend of peer A, and peers C and D are friends of peer B but are strangers of peer A. From peer B's local view, peer D provides better upload service than peer C. Peer D will have a good reputation from peer B. When peer A needs to choose which peer to cooperate with, it will ask for peer B's help. Peer B will give peer A the reputation value of peers C and D. Then peer A will choose peer D which has the high reputation value to cooperate with. But in the underlay network, the bandwidth and the latency between peer A and peer D are poorer than those between peer A and peer C. Therefore, it may affect the performance of the overlay network for peer A to choose peer D to cooperate with.

We also presented a round robin seeder unchoke scheme and fixed bootstrap data set as auxiliary mechanisms to prevent free riders from getting more resources.

3   Performance Evaluation

Inspired by [10], we design a discrete event driven simulator of BitTorrent based on OMNeT++ [11], INET framework, and OverSim [12]. We implement BitTorrent protocol including a tracker and an original BitTorrent client (OriBT), and an enhanced BitTorrent client (EnhBT) with our proposed incentive mechanism based on single-hop friends' recommendation and verification. We vary the number of peers from 50 to 300 with the different percentage of free riders or different types of client mixed. One initial seed is set to never sleep in any of the cases and the rest of the entering peers initially contain no pieces. We set all the

Table 3. BitTorrent Simulation Parameters

Number of total peers	50, 100, 300	Optimistic unchoke period	30 seconds
Number of initial seeds	1	Size of shared file	50MB
Peer arrival time	Poisson	Size of chunk	256KB
Number of unchoke neighbor peers	5	Downlink bandwidth for BitTorrent	1024Kbps
Regular unchoke period	10 seconds	Uplink bandwidth for BitTorrent	512Kbps
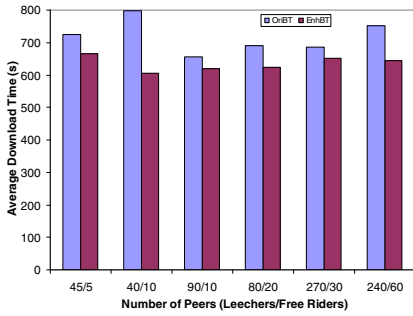
Fig. 3. Average download time of non-free-riders with different free rider fraction
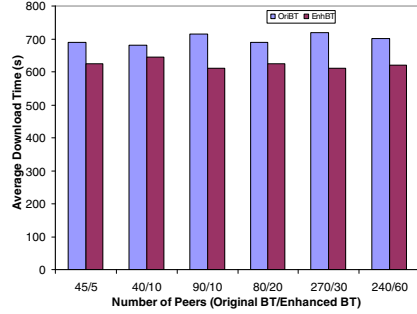
Fig. 4. Average download time of original BT vs. enhanced BT

peers having the same upload and download capabilities. The upload bandwidth of free riders is zero and download bandwidth is same as other ordinary peers. Table 3 shows the simulation parameters of BitTorrent.

Figure 3 shows the average down time of all non-free-riders in an original BT swarm and in an enhanced BT swarm mixed with 20% or 10% free riders separately. The average download time decreases in enhanced BT swarm with different free rider fraction. This means that the enhanced BT client could reduce the impact of free riding effectively.

Figure 4 depicts the average download time of the enhanced BT clients and the original BT clients with the different fraction of these two types of clients mixed. Enhanced BT clients maintain a lower average download time when run against original BT clients. This means that the enhanced BT client could improve the performance of the whole system.

4   Conclusion

In this paper we propose an incentive mechanism for BitTorrent based on single-hop friends' recommendation and verification. Simply by the single-hop recommendation and verification, the peer can find out more non-free-riders to optimistic unchoke instead of random unchoking. The chances of the free riders to get the optimistic unchoke slot is limited, and then the free data obtained by free riders is reduced, and thus the performance of the non-free-riders is improved. At the same time, the more recommendation a peer provides the more optimistic uchoking chances it gets. This incentive peers to recommend. It needs to upload a specific amount of accurate file data to qualify as a recommender once. Hence the novel mechanism also incentives peers to upload more file data. We also present a round robin seed unchoke scheme as an auxiliary mechanism to prevent the high-capability free riders from getting more seeds' capacity.

We plan to verify the effectiveness of our proposed incentive mechanisms by realistic deployments, and to improve the algorithm's effectiveness under extreme conditions by considering more limitations.

References

1. Ipoque Internet Study 2008/2009 (2010),
   `http://www.ipoque.com/resources/internet-studies/`
   `internet-study-2008_2009`
2. Locher, T., Moor, P., Schmid, S., Wattenhofer, R.: Free Riding in BitTorrent is Cheap. In: 5th Workshop on Hot Topic in Networks (HotNets-V), pp. 85–90. ACM Press, Irvine (2006)
3. Piatek, M., Isdal, T., Anderson, T., Krishnamurthy, A., Venkataramani, A.: Do Incentives Build Robustness in BitTorrent? In: 4th USENIX Symposium on Networked Systems Design and Implementation (NSDI), pp. 1–14. ACM Press, Cambridge (2007)
4. Bharambe, A., Herley, C., Padmanabhan, V.N.: Analyzing and Improving a Bit-Torrent Networks Performance Mechanisms. In: 25th Conference on Computer Communications (INFOCOM), pp. 1–12. IEEE Press, Barcelona (2006)
5. Jun, S., Ahamad, M.: Incentives in BitTorrent Induce Free Riding. In: ACM SIG-COMM Workshop on Economics of Peer-to-Peer Systems, pp. 116–121. ACM Press, Philadephia (2005)
6. Sirivianos, M., Park, J.H., Chen, R., Yang, X.: Free-Riding in BitTorrent Networks with the Large View Exploit. In: 6th International Workshop on Peer-to-Peer Systems (IPTPS), Bellevue, WA (2007)
7. Qiu, D., Srikant, R.: Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks. In: ACM SIGCOMM, pp. 367–378. ACM Press, Portland (2004)
8. Lian, Q., Yu, P., Yang, M., Zhang, Z., Dai, Y., Li, X.: Robust Incentives via Multi-Level Tit-for-Tat. In: 5th Workshop on Peer-to-Peer Systems (IPTPS), Santa Barbara, CA (2006)
9. Piatek, M., Isdal, T., Krishnamurthy, A., Anderson, T.: One Hop Reputations for Peer to Peer File Sharing Workloads. In: 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI), pp. 1–14. ACM Press, San Francisco (2008)
10. Katsaros, K., Kemerlis, V.P., Stais, C., Xylomenos, G.: A BitTorrent Module for the OMNeT++ Simulator. In: IEEE International Symposium on Modeling Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 1–10. IEEE Press, London (2009)
11. OMNeT++ (2010), `http://www.omnetpp.org`
12. Baumgart, I., Heep, B., Krause, S.: Oversim: A Flexible Overlay Network Simulation Framework. In: 10th IEEE Global Internet Symposium in conjunction with IEEE INFOCOM, pp. 79–84. IEEE Press, Anchorage (2007)

A Compact XML Storage Scheme Supporting Efficient Path Querying

Xiangyu Hu, Haiwei Zhang, and Xiaojie Yuan

Department of Computer Science, Nankai University, Tianjin, China
{huxiangyu,zhanghaiwei,yuanxiaojie}@dbis.nankai.edu.cn
http://dbis.nankai.edu.cn

Abstract. XML is becoming the de facto standard to store, exchange and publish information over the web, the need to develop efficient techniques for storing and querying XML documents has emerged. Two challenges in handling XML are its inherent verbosity and the complexity of its structure. A number of labeling schemes have been proposed to support fast XPath query processing, though they require even more storage cost. Compaction of XML documents has become an increasing important research issue. In this paper, we propose a new XML storage scheme called RRZip. In our approach the structure and content data of an XML document are stored separately: relative region labeling scheme are used for both compacting and querying the structure while the content data can be compressed by general-purpose compressors. Experiments show that proposed storage scheme is space and time efficient.

Keywords: XML, Compact Storage, Query Processing, Labeling Scheme

1 Introduction

eXtensible Markup Language (XML)[1] is emerging as de facto standard for information exchange and data representation on World Wide Web due to XML's inherent data self-describing capability and flexibility of organizing data. As the amount of XML data increases, it is becoming vital to be able to store and query this information.Many approaches have been proposed to provide robust storage and accelerate query processing. two common strategies of them are labeling schemes[2][3][4][5] and holistic twig join algorithms[6][7] ,which are both widely used in XML database systems. The first can be used to determine the relationship between any two nodes in an XML document in a constant time while the second to find all matched sub trees during one pass of the relevant nodes.

However, the inherently verbosity of XML led to the fact that the amount of information that has to be processed , stored and queried is often larger than that of other data formats, and labeling schemes will make things worse. Compression of large XML data not only reduce the cost of disk spaces, but it also speedup the overall query processing efficiency in database systems thanks to the reduced disk transfers necessary to access the XML document in compressed form.

In this paper we propose RRZip, a compact storage scheme that has the following desirable features: (1) achieves a comparable compression ratio with respect to gzip (2) supports efficient query processing on compressed XML data; and (3) it is very practical and can be collaborate with most state-of-art twig join algorithms.

This rest of paper is organized as follows. Section 2 introduces related work on XML labeling and compression techniques. In Section 3, our compact storage scheme is described in detail. In Section 4, experimental results are presented with comparisons to other existing approaches in this field. Finally, Section 5 concludes the paper.

2   Related Work

2.1   Numbering Schemes

In order to facilitate query processing for XML data, several labeling schemes have been proposed. Dietz[2] coding was the first numbering scheme based on region numbering. A pair of numbers (pre, post), which correspond to the pre-order and postorder traversal numbers of the node in the XML tree, is assigned to each node in this scheme. As each opening tag and closing tag of an element occur in pairs, three numbers (start, end, level) ,which correspond to the start position , end position of an element in document order and the depth of the node in the tree, is used in Zhang coding[3].It is easy to see that Zhang coding is equivalence to PrePost coding.

2.2   XML Compressors

As XML uses plain text representation for tree data, all general-purpose text compressors can be used as XML compressors natively. The first XML-conscious compressor is XMill[8], which is designed for exchanging and storing XML documents.The structure and content are separated in XMill which can increase the data similarity in each of them and allows to achieve better data compression rates.However, XMill does not allow direct querying of the compressed data. In XQzip[9], the duplicate structures are removed from the XML document to improve query performance by using an indexing structure called Structure Index Tree (SIT) and extra indices also utilized to accelerate querying.In [10], the authors adapted the XBW transform to derive a compressed searching and navigation tool for XML called the XBzip compressor. Recently, an ISX storage scheme is proposed by Raymond K. Wong[11], which is based on the balanced parenthesis encoding, representing the topology of an XML document. ISX supports fast navigational operations, efficient join operations. While CXDLS[12] ,based on ORDPATH labeling scheme compacts the structures of XML documents by exploding repetitive consecutive subtrees and tags in the structures.

3   Compact Representation of XML Data

3.1   The Relative Region Labeling Scheme

The RR labeling scheme is a variant of Zhang labeling scheme[3], which is a kind of region coding scheme and widely used in twig query algorithms. In original Zhang labeling scheme a tuple with three attributes (*StartPos*, *EndPos*, *Level-Num*) is used to represent the position of a node. An example of Zhang labeling scheme is illustrated in Fig. 1.

```
<bib>
  <book>
    <title>XML</title>
    <author>AC</author>
  </book>
  <book>
    <title>DB</title>
    <author>DG</author>
    <author>HI</author>
  </book>
</bib>
```
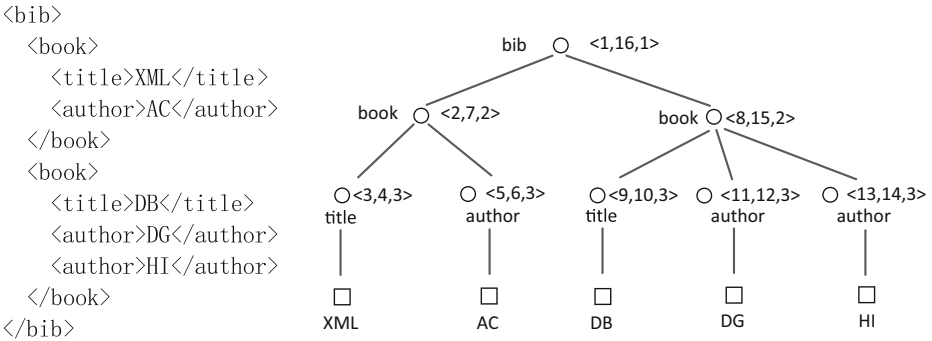
Fig. 1. A simple XML document with Zhang labels

Definition 1: A tag stream is a list of elements in document order that have the same tag name from an XML document. The operations over a tag stream can be: *next, eof and advance.* The operation next returns the heading element in the stream, eof returns true if the end of the stream has reached and the operation advance discards the heading element of the stream.

It is essential to provide a tag stream interface for XML query processing. The inverted index technology is often employed as storage engine in other region labeling schemes. In our approach an explicit tag stream is maintained for each distinct node tag. The offset between two contiguous nodes is used for representing the structural information, that is, for any node N_i (i>1) whose label is (S_i, E_i, L_i) in Zhang labeling scheme, and the label of the node $N_{(i-1)}$ before it is $(S_{(i-1)}, E_{(i-1)}, L_{(i-1)})$, a tuple with 3-field (*RelStartPos*, *RelEndPos* ,*Level-Num* where $RelStartPos = S_i - S_{(i-1)}$, $RelEndPos = E_i - E_{(i-1)}$ and $LevelNum = L_i$ is used to record position information. Whereas the first node of each tag stream remain its label in Zhang label scheme. For example the tag stream book have two nodes (2,7,2) and (6,8,2), while the tag stream author have three nodes: (5,6,3), (6,6,3) and (1,1,3).

Due to the fact that smaller integers are occurred more frequently than the larger ones in RR in our labeling scheme, prefixed Variable-Length Integers (VLI) technology is employed. As the *RelEndPos* of a node may be either positive or negative integer, an extra bit is needed to store the sign of the *RelEndPos*.

3.2   Compact Representation of XML

During XPath query processing structure is needed for navigation from one axis step to another, while content data are only needed when there is a prediction or final publish. Therefore separate the XML structure from its content data is very useful, because this separation typically lets queries only scan the structure, avoids unnecessary scanning of content data which will significantly reduce the access of disk and improve the query performance. As mentioned in the previous section 3.1, an explicit tag stream is maintained for each distinct tag and all nodes with the same tag name are clustered stored in document order, structural indices are necessary for accelerating random access and the indices are required to be small enough to be held in main memory.

In most case, an XML document is organized in a semantically meaningful manner. This means that the content data under the nodes with the same tag are likely to exhibit the same distribution bias. On the other hand the words under one tag are likely to have little intersection with words under another tag, or their distribution is very different. Grouping the content data under the same tag into a container and compressing different contains individually will allow the compressor to select a proper compression source model and finally improve the compression ratio.

The storage layer of the RRZip consists of three layers, namely, index layer, structural layer and content layer. The structural layer stores the position information of each node in a tag stream using a sequence of disk blocks whose sizes are 4 kilobytes. Each block contains contiguous arrays of tuples, with each tuple holding structural information about extra one node in an XML document. For sake of simplicity we also use a tuple to denote the node in the tuple when there is no ambiguous. A tag symbol table is maintained to record meta data of each tag, such as the name, the start offset of the block sequence and the total disk blocks used.

All tuples are aligned to byte boundary to reduce the cost of bit operation and improve disk performance, though it may also hurt the compression ratio. The size of a tuple can vary from at least three bytes to at most twelve bytes and one block can hold at least 340 tuples. The detailed description about a tuple is shown in Fig. 2.

Fig. 2. A Tuple in Structure Layer

The leading four bits in a tuple are used as tuple flags: Flag M indicates that one more byte is used to store the *LevelNum*. If it is set to 1, one and a half bytes are used and up to 4096 levels can be represented. Flag S indicates the sign of *RelEndPos*. If it is set to 1, the *RelEndPos* is a negative integer and otherwise

it is a positive integer. Flag C indicates this node has value and flag R are reserved for further usage. As the abstract *StartPos* and *EndPos* of a node can only be calculated by adding up the values of the previous nodes in the same tag stream, which is a heavy overhead for random accessing, a B+ tree index is built for each tag stream. The abstract *StartPos* of the first tuple in a disk block is served as the key and the disk block address is indexed by the B+ tree.

A straightforward method for linking the tuple to its content data is using global pointers, though it may cost extra four bytes for the nodes which are not empty elements. Another method is adopting the variable length field representation from the RDBMS technology which is more compact. In our approach for every block b in the tag stream there is a corresponding character sequence s, which is composed by the content items associated with the tuples in b in the same order with a delimiter χ to separate two adjacent items, where χ is a special character not occurring elsewhere in s. According to XML specifications from W3C, χ may be the character '<' or '>'. For the i-th tuple in a block, we can easily access its content data which is also the i-th item in the corresponding character sequence of the block.Benefited from the separation between structure and content data, the content data can be compressed individually by any general-purpose compressor.

3.3   The Space Cost for RR Labeling Scheme

In the section we will show the storage cost of the RR labeling scheme is linear to the number of the nodes and the detail upper bound.

THEOREM 1. Given an XML document X which has n nodes and t distinct tags, the upper bound space requirement of the structural layer is $4n + 0.048tn$ bytes. Especially when the maximum depth of X is less than 16, the upper bound space requirement is $3n + 0.04tn$ bytes.

PROOF (Sketch). Assume that the tag stream T_i has n_i nodes, all nodes need 12 bits to represent *LevelNum*, it is clear to see that at most n_i nodes need 4 bytes to be stored. At most n/128 nodes in T_i require 6 bytes because the minimal distance of any two nodes which is stored in 5 bytes should be 128 and the maximal distance of any two nodes will never greater than n. The upper bound space required by T_i is:

$$O(T_i) = 4n_i + \frac{6n}{128} + \frac{8n}{16,384} + \frac{10n}{2,097,152} + \frac{12n}{268,435,456} < 4n_i + 0.048n \quad (1)$$

And the upper bound space required by the document X is:

$$O(X) = \sum_{T_i \in T} O(T_i) = 4n + 0.048tn \quad (2)$$

The upper bound space requirement for the document whose maximal depth is less than 16 can be proved analogously.

4   Experiments

4.1   Test Preparation

The RRZip is implemented in C++ using libxml2 parser[14]. In this section, we compare performances of RRZip with other related implementations: gzip[13] (version 1.3), XMill[8] (version 0.8), XBZip[10] (version 1.0) and eXist[16] (version 1.4). All experiments were run on a DELL workstation with an Intel Core i7 920 processor (4 cores with hyper-thread enabled), 6GB of RAM and a 640G 7200rpm SATA hard disk. RRZip, gzip, XMill, XBZip were tested under Fedora with gcc 4.5.1 and eXist was tested under JRE 1.6.

The testing corpus represents a wide range of real-world XML applications and all documents in them are public available, including DBLP (856M), Shakespeare (7.52M) ,TreeBank (82M), XMark (111M). The xmlgen tool from XMark[15] is also used to generate XML files from 1M to 1G for scalability tests. We choose these documents because they have different characteristics.

4.2   Compression Ratio Comparisons

We measure storage requirements of RRZip to determine the power of out storage scheme on reducing the storage costs. We also compare these results with the storage costs of gzip, XMill and XBZip. The results of eXist are omitted for reason that it always requires more space than the original XML documents.

Fig. 3 shows the comparison of the compression ratio between different implementations. RRZip has a comparable compression ratio with gzip and other XML conscious compressors. Compared to XMill and gzip which are designed for archive and does not support any direct queries, XBZip supports a subset of XPath expressions. In fact, XBZip failed to compress the DBLP document in our experiments. We also generate a group of XMark documents whose sizes are from 1MB to 1GB and then compress them using RRZip and gzip. The results in Fig (b) shows the storage requirement of RRZip is almost linear to the source document files.

4.3   Query Performances

We investigate the query performance of RRZip and compare it with XBZip and eXist. All queries can be supported by RRZip and eXist while only a part of them can be processed by XBZip. TwigList Algorithm is employed in our implementations for processing complex twig queries. Tab. 1 shows these queries on XMark (Q1-Q8) and Shakespeare(Q9-Q15) datasets and present the pure query evaluation times. Each query was repeated 10 times and the average of them was recorded as the final results. These results confirm that RRZip can facilitate remarkable query performance.

(a)   Compression Ratio   (b)   Scalability
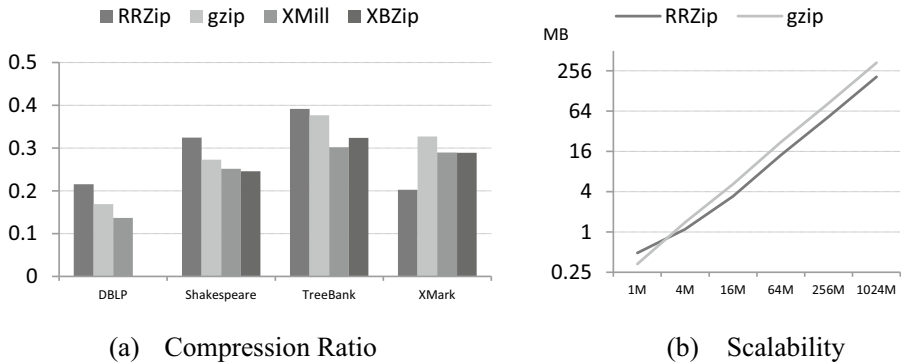
Fig. 3. Comparison of Compression Ratio

Table 1. Performance Evaluation Results on XMark

ID	Queries	RRZip	eXist	XBZip	Results
Q1	//regions/africa//parlist	375ms	177ms	-	294
Q2	//closed_auctions/closed_auction/price	260ms	194ms	409ms	9750
Q3	/site/people/person/name	101ms	127ms	452ms	1
Q4	/site/closed_auctions//emph	83ms	102ms	-	12206
Q5	/site/regions//item/description	116ms	111ms	-	21750
Q6	//people//person	120ms	65ms	-	25500
Q7	//africa//item	73ms	55ms	-	550
Q8	//africa//item[location="United States"]	159ms	96ms	-	398
Q9	/PLAYS/PLAY/ACT/SCENE/SPEECH	623ms	156ms	411ms	30933
Q10	/PLAYS//SCENE	60ms	63ms	-	750
Q12	//PLAY/ACT/SCENE//LINE/STAGEDIR	197ms	159ms	-	618
Q13	//PLAY//STAGEDIR	189ms	95ms	-	6258
Q14	//PLAY/ACT	28ms	57ms	338ms	185
Q15	//LINE[STAGEDIR="Aside"]	308ms	176ms	462ms	208

5   Conclusion

We have described a new approach called RRZip, which can significantly reduce
the storage requirement of XML data and support efficient query processing
directly on compressed formats, which is achieved by a combination of XML
relative region labeling scheme and content compression. Our experiments veri-
fied that RRZip (1) achieves comparable compression ratios with respect to gzip
and fast mutli thread compression/decompression, and (2) achieves extremely
competitive query performances on the compressed XML data. As future work,
we plan to extent our work with dynamic updating support.

References

1. W3C, Extensible Markup Language (XML), XML Path Language (XPath), XQuery 1.0: An XML Query Language, `http://www.w3.org/TR/`
2. Grust, T.: Accelerating XPath location steps. In: Proceedings of SIGMOD 2002 (2002)
3. Zhang, C., Naughton, J., Dewitt, D., Luo, Q., Lohman, G.: On supporting containment queries in relational database management systems. In: Proceedings of ACM SIGMOD (2001)
4. Tatarinov, I., Viglas, S., Beyer, K., Shanmugasundaram, J., Shekita, E., Zhang, C.: Storing and Querying Ordered XML Using a Relational Database System. In: SIGMOD (2002)
5. O'Neil, P., O'Neil, E., Pal, S., Cseri, I., Schaller, G., Westbury, N.: ORDPATHs: Insert-friendly XML node labels. In: SIGMOD, pp. 903–908 (2004)
6. Bruno, N., Koudas, N., Srivastava, D.: Holistic twig joins: Optimal XML pattern matching. In: SIGMOD 2002, Madison, Wisconsin (2002)
7. Qin, L., Yu, J.X., Ding, B.: *TwigList*: Make Twig Pattern Matching Fast. In: Kotagiri, R., Radha Krishna, P., Mohania, M., Nantajeewarawat, E. (eds.) DASFAA 2007. LNCS, vol. 4443, pp. 850–862. Springer, Heidelberg (2007)
8. Liefke, H., Suciu, D.: XMill: an Efficient Compressor for XML Data. In: SIGMOD (2000)
9. Cheng, J., Ng, W.: XQzip: Querying Compressed XML Using Structural Indexing. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 219–236. Springer, Heidelberg (2004)
10. Ferragina, P., Luccio, F., Manzini, G., Muthukrishnan, S.: Compressing and searching XML data via two zips. In: WWW 2006, pp. 751–760. ACM, New York (2006)
11. Wong, R.K., Lam, F., Shui, W.M.: Querying and maintaining a compact XML storage. In: WWW (2007)
12. Alkhatib, R., Scholl, M.H.: Compacting XML Structures Using a Dynamic Labeling Scheme. In: Sexton, A.P. (ed.) BNCOD 2009. LNCS, vol. 5588, pp. 158–170. Springer, Heidelberg (2009)
13. gzip, `http://www.gzip.org/`
14. XML libxml2, `http://xmlsoft.org/`
15. Schmidt, A., Waas, F., Kersten, M.L., Carey, M.J., Manolescu, I., Busse, R.: Xmark: A benchmark for xml data management. In: VLDB, pp. 974–985 (2002)
16. eXist-db Open Source Native XML Database, `http://exist.sourceforge.net/`

Self-supervised Learning Approach
for Extracting Citation Information on the Web

Dat T. Huynh and Wen Hua

School of Information Technology and Electrical Engineering
The University of Queensland, Australia
{tandat.huynh,w.hua}@uq.edu.au

Abstract. In this paper, we propose a framework for automatically training a model to extract citation information on the web. Constructing manually labeled training data to learn an extraction model is tedious, time consuming and difficult to be applied to several styles of citations with different types of entities. To eliminate the requirement of manually labeled training data, we exploit a knowledge base of citation domain and web search to derive labeled training data automatically. Our experiments show that the combination of knowledge base, heuristics and statistical methods can automate the extraction process and achieve good performance.

1 Introduction

Citation extraction is a process of extracting field values e.g authors, title, publication venue, page, year from citation strings. Designing an application that can automatically recognize and parse citations scattered over the internet is not a trivial task because of several reasons. Firstly, people write citations in several styles with different orders of field values and punctuation marks. Moreover, since field values are expressed in a textual representation, traditional wrapper-based methods ([6], [2]) cannot be applied to the inputs formatted differently in HTML. Instead, citation extraction is related to the problem of segmenting texts to extract data values in them. A dominant approach for this problem is the deployment of statistical methods such as Hidden Markov Model (HMM) ([3]) and Conditional Random Fields (CRFs) ([9]).

However, these learning based methods require a large amount of labeled training data. Obtaining such a training dataset to extract several types of entities from the web or entities in citations written in any style may cost a lot of time and labor work. In this paper, we propose a framework that automatically constructs learning data from the web and trains a parser to extract information of citations. In our framework, we exploit a structured database of citation records to construct a rich training dataset automatically from the web, which is then utilized to train a statistical extraction model for parsing citations.

The structure of this paper is organized as follows. In section 2, we discuss related works in the literature of citation extraction. Next, section 3 formally

defines the problem and presents our proposed method to automatically construct training data from the web for a statistical learning method. After that, experiments and evaluations are illustrated in section 4. Eventually, section 5 concludes the paper and suggests some future works.

2   Related Works

In general, existing research on the problem of citation extraction can be categorized into rule-based approach and learning-based approach. Rule-based approach exploits knowledge of a particular domain to describe data of interest. From a knowledge base of citations, several templates, rules and extractors are generated to extract information of citations. This idea can be found in IN-FOMAP system ([7]) which employed an ontology to match predefined templates with citations written in six fixed reference styles. As a consequence, the ability to adapt to the diversity of reference styles is still a big challenge for this research work. Flux-CiM ([4]) is another study that exploits a reference table in citation domain to extract metadata of citations. This method used punctuation marks to split each citation string into text blocks and defined fitness functions to estimate the probability of blocks to be labeled as fields in the reference table. Although this method is robust in extracting citations in different styles, it requires that the vocabulary set of a field in the reference table must be large enough to cover a representative portion of the domain of interest. Therefore, any bias on the size of vocabularies in a reference table will directly affect the probability of a candidate text block to be labeled as a field in the reference table, thus influencing the quality of the extraction process.

In learning-based approach, previous works formulated citation extraction as the problem of labeling a sequence of words. One solution to assign labels for tokens is to view the problem of labeling tokens as the problem of classification in which an extraction model must determine whether a token is assigned a particular label or not ([8]). Moreover, to capture the dependency between the labels and features of adjacent words in strings, some graphical extraction models were proposed, such as Hidden Markov Models (HMMs) ([14], [3], [1]) and Conditional Random Fields (CRFs) ([9], [12], [5]). Currently, CRFs-based methods are state-of-the-art and outperform all previous methods in both theory and experimental evaluations ([13]).

Those learning-based methods had good adaptability and performed well in experiments but their training datasets were manually built. The authors in [11] tried to reduce the dependence on manually labeled training data by exploiting an existing structured database to augment some new features. However, it still needs some user-provided training data. Recently, [16] has proposed an unsupervised method with CRFs. In [16], a training dataset is directly generated from a reference table by concatenating attribute instances in the table according to a total order, which is inferred from input text. Although this method does not require training data, it has some limitations. Firstly, it is very common to have more than one order of attributes in a single dataset. Moreover, since field values in a reference table can be represented in a different format with field values

occurring in input text, the CRFs-based model may not learn enough features to obtain high performance if the reference table and the input texts come from different sources. In addition, this method has bad performance in running time because it executes the inference step and training step each time it performs a new extraction.

In general, statistical learning methods are flexible and robust, but they still have some limitations. Therefore, our proposed method is to exploit a knowledge base and generate training data automatically from the web. This method helps us to overcome the disadvantages of statistical learning method, obtain high performance, and automate the process of information extraction.

3   Proposed Method

We refer to a *citation* or a *reference* as a string representing field values of an academic publication. A typical representation of a citation includes information of authors, title, book title or publication venue, pages, date, volume, and some other information of a publication. These fields are usually separated by punctuation marks. Formally, each citation string S can be represented as $S = \{Field_1 Delimiter_1 Field_2 \ Delimiter_2 Field_3 Delimiter_3...\}$, where $Field_i$ are field values of citations, and $Delimiter_i$ are symbols defined by any character other than A..Z, a..z, or 0..9 to split a citation string into a sequence of *tokens*. Citation strings are written in particular *styles* which define the order of fields and delimiters. Some citation strings include complete information such as author, title, publication venue, date, location, and publisher while others use only a subset of these fields. Moreover, field values can be written in different aliases or abbreviations in different citations.

We are interested in automating the process of building large labeled training dataset and then incorporating statistical learning methods to automatically extract field values of citations written in any style on web pages. Figure 1 describes the overall architecture of our framework. We firstly use information of citation records from DBLP to search and extract citation strings from the web, and then apply heuristics to annotate the field values in these citation strings. We show that the combination of heuristic and statistical methods helps us to automate the extraction process. In the following sections, we will describe the modules of our framework in detail.

3.1   Extract Citation Strings from Web

This module is to extract citation strings representing each entry in DBLP from the web. Firstly, we use publication title of each DBLP record to search on the web by using a keyword-based search engine such as Google or Bing. Then, a set of hyperlinks on returned web pages is extracted by using regular expressions. These hyperlinks are used to download web pages containing information of those publications. After that, an HTML parser is built to analyze the structure of those web pages and find all text nodes specified by common HTML tags, such
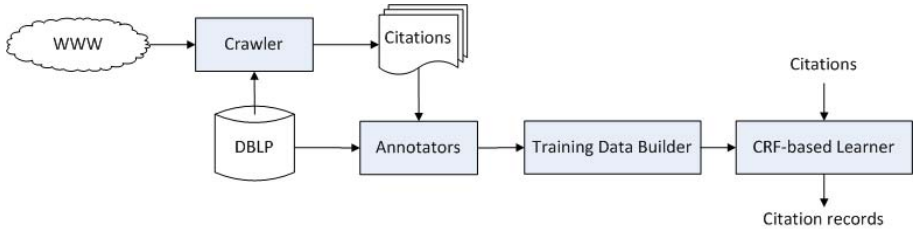
Fig. 1. The flow chart of our system for citation extraction

as $< li >$, $< td >$, and $< p >$. We extract all text paragraphs which contain the title of each publication record in our knowledge base.

Since not all text paragraphs containing the title are citation strings for the publication records, we define some criteria to pick out text paragraphs T that indeed match the current searching record R. Firstly, the text paragraph T must contain at least one word of each author name in the record R. Secondly, all words in the publication title in the record R must appear in the text paragraph T. Thirdly, either information of year or page numbers must appear in T. Besides above criteria, we also define some simple rules to discard noise data such as the Bibtex and EndNote files, which may be returned in searching results. These extracted citation strings are passed into annotator modules to recognize field values of record R in these citations.

3.2 Label Citation Strings

The main purpose of this module is to find text segments in each citation string which best match with field values of the structured record in DBLP. This task is complicated because exact string matching criteria might lead to too few matches, while soft criteria might generate a few wrongly labeled strings which will severely impact the accuracy of the CRFs model trained on these training data. Moreover, field values in DBLP can be represented in some noise forms on the web. For example, authors' names, publication venues, or dates can be written in different representations or abbreviations, and some field values, e.g years or conference names, might occur multiple times in citation strings.

Formally, given a record R in DBLP and a citation string S, we need to find a set of text segments in S that approximately match with the field values in record R. It is necessary to define a soft scoring function *SoftScore(s, R, A)* which returns a value in [0, 1] and measures the similarity between a candidate text segment s in S and a field value of the attribute A in the record R. The best set of segments S^* that matches with the record R can be defined by $S^* = \arg\max_{s:\ s\ valid\ for\ S} \sum_{s \in S} SoftScore(s, R, A)$.

In our implementation, we define a set of rules to generate different aliases for each attribute A in DBLP. The aliases of a field value in each record R will then be used to find a text segment s in citation string S which best matches with that

value. We firstly match the tokens in the aliases of field values with each citation string to find a matched segment s and annotate its tokens. After that, since not all fields can be found a exact matching, we continue to exploit structural information encoded by delimiters in citation strings to label the remaining field values. We generate from each citation string a list of groups of consecutive tokens bounded by any two delimiters. Then the aliases of remaining fields are approximately matched with each group of tokens. Each group is considered as a candidate segment s and we calculate a score *SoftScore(s, R, A)* for aliases of A in current record R. In our implementation, we measure the similarity between a candidate segment s and an alias by using edit distance ([10]). We will choose the best candidate segment whose similarity measure to one of the aliases of a field greater than a threshold θ.

3.3   CRF-Based Extraction Model

Our extractor utilizes CRFs to label tokens in input strings. More detailed information on this algorithm can be found in [9]. In this section, we concisely provide an overview of CRFs and describe features we employ in our system.

   CRFs is a discriminative learning model which directly computes the conditional probability distribution of a sequence of labels y given a particular sequence of input tokens x. This is contrary to generative models, such as HMMs, which compute a joint probability distribution over both output label sequence and input token sequence. In CRFs model, the likelihood of a label sequence given an input token sequence is defined by a set of features of tokens in the sequence which capture the relations between labels and tokens as well as relations between labels in the sequence. In our implementation, we employ Mallet[1] library to build our extractor and the features for CRFs include orthographic cases, numbers, punctuation, dictionary of person names, and word features.

4   Experiments

In our experiments, we test our method on PersonalBib dataset made up of 400 citations of journal articles manually labeled and used in [11]. We randomly choose 3,800 bibliography entries in the following schema: [*Title, Author, Journal, Volume, Pages, Year*] from DBLP repository. We firstly perform the algorithms in section 3 to crawl from the web and obtain 27,489 citation strings. Since training dataset directly affects the performance of extraction model, we assess the accuracy of automatic labeling module in our framework. We randomly choose 100 labeled citations generated by our method and manually count the number of tokens labeled correctly for each field. Table 1 reports the results of this experiment. Moreover, we verify how the size of knowledge base affects the overall effectiveness of extraction process by varying the number of records that we use from DBLP. Figure 2a illustrates the results when we test our framework

---

[1] http://mallet.cs.umass.edu

Table 1. Performance of the module of labeling citations

Field	Precision	Recall	F1
Title	100.00%	100.00%	100.00%
Author	98.51%	96.03%	97.25%
Journal	99.46%	94.16%	96.73%
Volume	98.92%	85.98%	92.00%
Pages	100.00%	99.58%	99.79%
Year	100.00%	99.00%	99.50%

on PersonalBib dataset. It can be seen that the performance of our method obtains over 80% and 90% when the size of the knowledge base exceeds above 80 and 2,000 entries, respectively. Meanwhile, figure 2b describes in detail the values of precision, recall and F1 measures for each of attributes in PersonalBib dataset when we build our extraction model from 3,800 DBLP entries. It can be concluded that rich knowledge bases for a domain should be exploited to achieve high performance in learning a statistical model for information extraction.

(a)   (b)
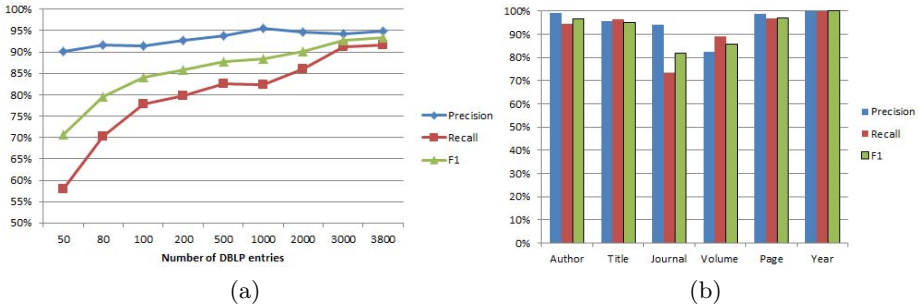
Fig. 2. Performance of our method

The idea of using a structure database to reduce the size of manually labeled training data was actually employed in [11]. However, information of DBLP in their work was used to augment some dictionary-based features for learning their extraction model. Their extraction process still requires some user-provided training data, similar to what happens with a standard learning model, while our method does not require any intervention of human in building training data. Meanwhile, the experimental results of [11] on PersonalBib dataset showed that the overall F1 measures on three fields of authors, journal and title are 69%, which is lower than ours when the number of used DBLP entries are above 80.

We also compare the performance of our method to the results of a recent study of [16] (referred to as U-CRF), which outperformed the work of [11] on PersonalBib dataset. To compare with U-CRF, we re-implement the algorithm of [16], then employ our 3,800 DBLP entries as a reference table. The comparisons of F-measure and running time for extraction between our method and U-CRF

(a) F1 measure (b) Running time
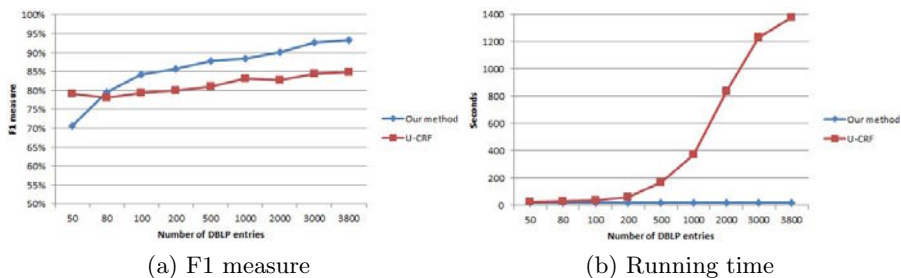
Fig. 3. Comparisons between our method and U-CRF

on PersonalBib dataset are presented in figure 3. In general, the F-measures of both methods become better when we increase the size of DBLP entries. However, U-CRF method makes less improvement. It is shown in figure 3a that our method outperforms U-CRF when the number of DBLP entries are greater than 80. This is mainly due to the fact that not all fields in citation strings of PersonalBib dataset are in the same order, which is inconsistent with the "single order" assumption of U-CRF, while training data in our method is derived from the web and it captures citation strings in different styles.

Moreover, it can also be seen from figure 3b that the processing time of our method is shorter than that of U-CRF. To explain this, we note that each time the method of [16] carries out a new extraction process, it requires a new model to be learned from test instances. As a consequence, the processing time of their method dramatically increase when we increase the size of reference table.

5 Conclusion

We have presented our framework for extracting citation information from the web. We have shown that the overlapping between different sources such as structured databases (e.g DBLP) and the web can be exploited to construct training data automatically. In the experiments, we have demonstrated the effect of the size of knowledge base on the performance of CRFs-based extraction model in our framework. We have shown that our method obtains good performance in practice and is better than that reported in [11] and [16].

Currently, training data in our method is generated for the fields defined in a knowledge base. Therefore, the extraction model can only label citations with those fields. Meanwhile, other useful field values may occur in citation strings on the web, such as locations or universities, but the knowledge base may not contain. One direction is to exploit the similar structures of citations to group these field values into clusters. Then they may be labeled by using a statistical method such as [15]. That is one of future works which we are investigating.

References

1. Agichtein, E., Ganti, V.: Mining reference tables for automatic text segmentation. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 20–29 (2004)
2. Arasu, A., Garcia-Molina, H.: Extracting structured data from web pages. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp. 337–348 (2003)
3. Borkar, V., Deshmukh, K., Sarawagi, S.: Automatic segmentation of text into structured records. In: Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, pp. 175–186 (2001)
4. Cortez, E., da Silva, A.S., Gonçalves, M.A., Mesquita, F., de Moura, E.S.: A flexible approach for extracting metadata from bibliographic citations. Journal of the American Society for Information Science and Technology 60, 1144–1158 (2009)
5. Councill, I.G., Giles, C.L., Yen Kan, M.: Parscit: An open-source crf reference string parsing package. In: International Language Resources and Evaluation. European Language Resources Association (2008)
6. Crescenzi, V., Mecca, G., Merialdo, P.: Roadrunner: Towards automatic data extraction from large web sites. In: Proceedings of the 27th International Conference on Very Large Data bases, pp. 109–118 (2001)
7. Day, M.-Y., Tsai, R.T.-H., Sung, C.-L., Hsieh, C.-C., Lee, C.-W., Wu, S.-H., Wu, K.-P., Ong, C.-S., Hsu, W.-L.: Reference metadata extraction using a hierarchical knowledge representation framework. Decision Support System 43, 152–167 (2007)
8. Han, H., Giles, C.L., Manavoglu, E., Zha, H., Zhang, Z., Fox, E.A.: Automatic document metadata extraction using support vector machines. In: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 37–48 (2003)
9. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289 (2001)
10. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10(8), 707–710 (1966)
11. Mansuri, I.R., Sarawagi, S.: Integrating unstructured data into relational databases. In: Proceedings of the 22nd International Conference on Data Engineering, pp. 29–40 (2006)
12. Peng, F., McCallum, A.: Information extraction from research papers using conditional random fields. Information Processing and Management 42, 963–979 (2006)
13. Sarawagi, S.: Information extraction. Foundation and Trends in Databases 1(3), 261–377 (2008)
14. Seymore, K., Mccallum, A., Rosenfeld, R.: Learning hidden markov model structure for information extraction. In: AAAI 1999 Workshop on Machine Learning for Information Extraction, pp. 37–42 (1999)
15. Venetis, P., Halve, A., Madhavan, J., Pasca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. Proceedings of the VLDB Endowment (2011)
16. Zhao, C., Mahmud, J., Ramakrishnan, I.V.: Exploiting structured reference data for unsupervised text segmentation with conditional random fields. In: Proceedings of the SIAM International Conference on Data Mining, pp. 420–431 (2008)

Complete-Thread Extraction from Web Forums⋆

Fanghuai Hu, Tong Ruan, and Zhiqing Shao

Department of Computer Science and Engineering, East China University
of Science and Technology
xiaohuqi@126.com, {ruantong,zshao}@ecust.edu.cn

Abstract. This paper proposes an effective algorithm which can auto-
matically extract all meta-information of threads from various forums.
The algorithm contains two steps: thread extraction from board pages
and detailed information extraction from thread pages. In the thread
extraction step, the board pages are divided into five types according to
their structure, and corresponding extraction algorithms and models are
suggested. In the second step, an effective method is applied to identify
the content of the origin post, other un-extracted fields of the origin post
which are always located around the content are matched by regular pat-
terns, and a model is trained to extract the reply posts. The experiment
shows that the proposed algorithm is accurate and effective.

1 Introduction

An Internet forum is an online discussion site where people can hold conversa-
tions in the form of posted messages. Many forums have been developed by some
famous software such as 'Discuz!' and 'SMF', but the rest may be developed by
different programmers and have ad hoc structures, which makes it a difficult
task to automatically extract information from randomly selected forums.

In this article we propose an effective algorithm to extract all meta-information
of threads from different kinds of forums. The main advantages of our algorithm
are that, it is adaptable to different forums, the task can be accomplished without
any human interaction after the models are trained, and all the meta-information
of a thread can be retrieved.

2 Related Works

Former works focused on forum IE were as follows. Cai et al.[1] built an intelligent
forum crawler called iRobot, which first used some sample pages to generate
the sitemap for each forum, and then selectd the optimal crawling path in a
semi-automated method way. Yang et.al. [5] combined page-level with site-level
knowledge to to the IE task, and then employ Markov logic networks to integrate

---

userful evidence by learning their importance automatically. Wang et al.[11] proposed an automatic approach to exploring appropriate traversal strategies to direct the forum crawling. However, all the three algorithms were built on the sitemap, which were not easy to generate without the help of manual work.

A board forum crawling method was proposed in [2] to crawl web forums: it first found all board pages, and then got all thread pages from the board pages. However, their experiments showed that the recall of the algorithm was only about 65%. Zhang et al.[3] proposed an approach for forum extraction based on the observations from how humans recognize information, but they only roughly get the information area and did not get the concrete information fields.

'Juicer' [4] was developed to extract meta information of each thread from board pages. The paper found out that there are three kinds of board pages: Table-type, Div-type, and Other-type. However, we claim that the classification is not concrete enough to deal with the diverse structures of forums, and they can only get part of the information fields from board pages since they didn't discuss how to retrieve the other information from the thread pages.

There are some other works about IE from forums, such as analyzing the structures of forums [6,7] and mining threaded forum information [8,9,10].

3   Problem Definition

In this section, we fist give a number of preliminary definitions which will be used throughout the paper, and then define the forum thread extraction problem.

1. **Listing HTML tag.** Listing HTML tags, usually including $< table >$, $< tr >$, $< li >$ and $< div >$, are used to arrange a series of regular data records. Threads in board pages are arranged by them in this article.
2. **XPath.** Similar to XML, we define the XPath of HTML here. Every node appears in the XPath with its tag name and sequence number in all its sibling nodes (For example, '/HTML 1/BODY 2/DIV 5'):
3. **Forum.** A forum is a triple $f := (hp, BP, TP)$, where hp means the home page , BP is the set of board pages, and TP is the set of thread pages.
4. **Post.** A post is a structure $p := (url, title, author, time, views, posts, content)$, consists of a URL, a title, an author, a posting time, a view count, a post count, and a content.
5. **Thread.** A thread is a pair $t := (op, RP)$, where op is the origin post and RP is a set of reply posts. Each t corresponds to an element in TP, and rp is an element of RP.
6. **Header.** A header of a board page is a style list to arrange threads. Usually it consists of particular forms of thread field names. If a header appears in a board page, typically it will be followed by a series of arranged threads.

Therefore, the process of forum information extraction can be described in terms of three steps: given the hp of a forum f, the first step is to find the BP of f from hp. Then, get the TP by analysing each bp of BP. Finally, from each element in TP, extract the detailed thread information. This paper will focus on the last two steps.

4 Thread Extraction from Board Pages

Our algorithm consists of two main steps: thread extraction from board pages and detailed information extraction from thread pages, which will be described in detail in this section and Section 5 respectively. The input is a URL of a board page. Match the header first, and detect the thread list if no header is found, then extract thread from the board page. A series of threads with URLs and titles at least are passed to the the next step. Then, step 2 takes the extracted incomplete threads as input, extracts the origin post and reply posts in turn. The output is the set of corresponding complete threads.

4.1 Board Page Structure Analysis

After a careful study of various forums on the Internet, we summarized five types of forums according to their board pages' structure:

- Single-Tabled-List (STL): Threads are in a large table, each thread in a row.
- Multi-Parallel-Tabled-List (MPTL): Threads are in several parallel tables, each thread in a table or a cluster of threads in a table.
- Multi-Non-Parallel-Tabled-List (MNPTL): The threads of this type are in nested tables.
- Non-Tabled-Compact-List (NTCL): These forums use other listing HTML tags instead of $< table >$ tags, and the threads are arranged by headers too.
- Non-Table-Incompact-List (NTIL): $< table >$ tags are not used in this type either. The threads are not regularly arranged by headers, and there is little information in the board pages except for the title field and the URL field.
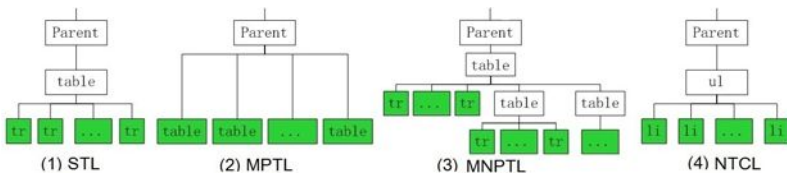
Fig. 1. DOM trees of different kinds of board pages

Figure 1 shows partial DOM trees of the first four types: each colour filled node corresponds to a thread (or a cluster of threads in MPTL type).

4.2 Thread Extraction

Header Detection. For the first four types, there are headers in the board pages, and all the subsequent threads follow its style. Later experiments will show that the result of forums with headers are much better. Therefore, we have

to detect the header first. Headers from different forums always have fields such as 'title', 'author', 'views', 'posts', and so on. We consider the header selection as a classification problem, only the header node is positive, and others are all negative. We train an SVM[13] classification model, the features we used are: (1) HTML tag type. Headers are usually contained in specific HTML tags such as $< div >$ and $< table >$. (2) Word in HTML tag. There are some specific forms for each field of header, for example, the forms of the title field are 'subject', 'title', 'topic', 'article', 'thread topic' in different forums. (3) The depth of child nodes. Generally, the header nodes donot contain any deeper child nodes.

Thread List Detection. For the last type, NTIL, as we do not know which type of tag is used as list tag, the thread list must be detected first. (1) For each node in DOM tree,if the node is a hyperlink node and SL(node-text) > 10 then add the node to the record list; (2) For each node in the record list, modify the XPath of the node and count the number of each group of nodes with the same XPath in the record list; (3) select the group with the maximum number, and use their common parent node as the thread list. The modification step is to remove the sequence number of the last listing HTML tag (if it exists). It is worth noting that we do not limit which kind of tag is the listing HTML tag.

Thread Extraction. After headers (if any) or the thread list (for NTIL) are detected, the thread URL and title can be extracted, and some other fields can also be extracted in most cases. As all threads of a board page which have a header are arranged by the header format, thread extraction is very simple. However, for NTIL type, it is more difficult as threads are arranged casually. After a careful study of many forums of this kind, we have found some heuristic rules to help in the extraction, for example: the title and the author are in hyperlink tags as there are thread pages link to them.

5   Detailed Information Extraction from Thread Pages

So far, the URL and the title of every thread have been extracted from the board pages, and some other fields have been extracted, too. The following step is to extract the un-extracted fields of the origin post and all content of reply posts from the thread pages. A thread page contains all the information of the including thread, but there is some other noisy information such as navigation bars, advertisements, and so on. By analysing a large number of thread pages from different kinds of forums, we find that all the fileds of a post are clustered, form a box. The origin post is always on the top of the reply posts.

5.1   Extracting the Origin Post

As the origin post is most important, all of its field should be extracted. In order to extract the *op*, we have to locate the *op* cluster first. As already mentioned,

the *title* is already extracted from the board page. And we find that the *content* of the *op* has several peculiar statistic characteristics which are very similar to the heuristic rules described in[14]. Therefore, we use the same algorithm to help finding the *content*, and only change the experimental parameter K_P to 8.

After the content is detected, other fields are much easier as they are near the title and the content. Moreover, they have their own specific patterns. Therefore, we only have to use many regular expressions to match them respectively.

5.2   Extracting the Reply Posts

Now we elaborate extracting the content of the reply posts, and also model the extraction process as a classification problem. In a thread page, the *rp* nodes are positive, and others are negative. The following features are used: (1) HTML tag type. Content are usually contained in container HTML tags [14] such as $< div >$ and $< table >$. (2) The SS of the node's XPath with the XPath of the content of *op*. The reply post are paralleled with the origin post in most case, at least in the neighboring levels. Therefore, the content fields of *rp* and *op* are similar. (3) Quantity of word. In general, the content contains more words than others. (4) Quantity of tag. In most case, there are always few tags in the content field.

6   Experiment

In order to evaluate the performance of the proposed algorithm in different forums, we have collected 100 forums requested by an IT service company for experimental data. The topics of these forums covering car, computer technology, entertainment, advertisement, tourism, education, and so on. Some of these forums are developed by famous software such as 'Discuz!', and others are developed by ad hoc technologies. Here we use the standard metrics, *recall* and *precision*, to evaluate our algorithm as most IE researchers do.

6.1   Header Detection Model Evaluation

To find out how many different forums are sufficient to train a accurate model, we chose 50 forums as training data and other 50 as testing data. The number of training data incrementing from 10 to 50, with a increase rate of 5. The result is show in Fig. 2. When the number of training data increase to 30, the recall reaches about 96%, and it almost does not increase when adding more training data. But even when 50 training forums are used , one forum's header has not been detected yet, because the header is really strange. Therefore, we conclude that 30 forums are enough for training a good model.

6.2   Result of Thread Extraction from Board Pages

In board page extraction, if the URL and the title of a thread are correctly extracted, we say the thread is successful extracted. The results of other fields
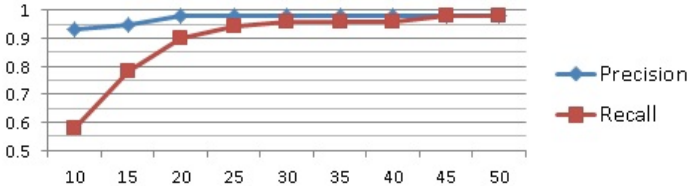
Fig. 2. The precision and recall of header extraction model trained by differt number of training data

Table 1. Result of thread extraction from board pages

Type	Quantity	Correct	Incorrect	Recall(%)	Precision(%)
STL	1262	1253	23	99.3	97.9
MPTL	1022	1006	41	98.4	96.1
MNPTL	836	807	48	96.5	94.3
NTCL	920	908	23	98.7	97.5
NTIL	518	443	132	85.5	77.0

will be evaluated in the thread page extraction step. Here we only choose 20 forums(4 for each type) to evaluate as the amount of threads of each forum is huge. For each selected forum, we extract the threads of the first five pages of an arbitrary board, and check the results manually, the result is shown in Table 1. From the table, we can find that the rulst of forums with headers are much better. The recall and the precision for the NTIL type is only 85.5% and 77.0%. Luckily, most forums belong to the first four types.

The reason for the missing threads is that some sticky threads are far from other threads, and we cannot scan and recognize them. The redundant threads are always some advertisements close to the thread list.

6.3   Result of Origin Post Extraction

In this section, we will discuss the extraction results of all other fields of the origin post, the total number of test threads is the correctly extracted threads in board page extraction. Figure 3 gives the result, the following can be concluded: (1) Both the recall and the precision of the four types of forums with headers are much better as many fields can be extracted correctly from board pages. (2) The results of the views and the posts are almost the same as they always appear together. The reason for the inaccuracy is some sticky threads have no views and posts, and even all threads do not have posts or views for some forums. (3) As the formats of posting time are distinctive, it is easily to extract the post time. Therefore, almost all of them are extracted correctly. (4) The result of the content are almost equivalent over the five type, because the same method is used.
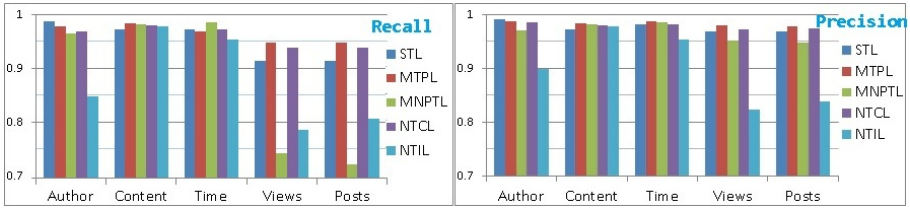
Fig. 3. The precision and recall of fields extracted from thread pages
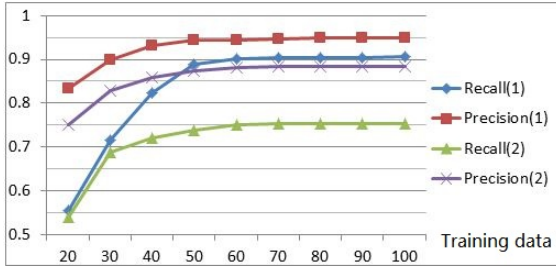
Fig. 4. The precision and recall of the *rp* content extraction model, (1) is the result of the model traind by thread pages from different forums, and (2) is the result of the model when thread pages from only one forum is used as training data

6.4   Reply Post Extraction Model Evaluation

Gradually increased number of manually labeled training data is used to train the model too, from 20 to 100. Moreover, we train two models by thread pages from only one forum and thread pages from different forums. The testing data consists of 50 thread pages from different forums, 1452 reply posts in total. The result is shown in Fig. 4. It illustrates that only use thread pages of one forum is not competent; No matter how many training forums are used, the recall will never be better than 75%. When using mutiple forums, the recall can reaches about 95%, and we also conclude that 50 forums are sufficient.

7   Conclusions

In this paper we discuss complete-extraction from forums. We divide the extraction process into two steps, thread extraction from board pages and detailed information extraction from thread pages. Structure similarities of board pages are found from different forums, according to which forums are divided into five types: STL, MPTL, MNPTL, NTCL, and NTIL. The header is significant for the thread extraction process, therefore, a model is trained to detect the headers of board pages, and we prove that only about 30 different forums are required to train an accurate model. Moreover, a useful method is utilized to extract

the content of the origin post, and an accurate model is trained to extract the content of the reply posts. The experiment shows our method is effective: we successfully extract all meta-information of threads from more than 100 forums, and there is no requirement for manual interaction in the process.

References

1. Cai, R., Yang, J.M., Lai, W., Wang, Y.D., Zhang, L.: iRobot: An Intelligent Crawler for Web Forums. In: Proceeding of the 17th International Conference on World Wide Web, pp. 447–456. ACM (2008)
2. Guo, Y., Li, H., Zhang, K., Zhang, G.: Board Forum Crawling: A Web Crawling Method for Web Forum. In: 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 245–748. IEEE Computer Soc. (2006)
3. Zhang, Q., Shi, Y., Huang, X.J., Wu, L.D.: Template-independent Wrapper for Web Forums. In: 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 794–795. ACM (2009)
4. Guo, Y., Wang, Y., Ding, G.D., Cao, D.L., Zhang, G., Lv, Y.: Juicer: Scalable Extraction for Thread Meta-information of Web Forum. In: Chen, H., Yang, C.C., Chau, M., Li, S.-H. (eds.) PAISI 2009. LNCS, vol. 5477, pp. 143–148. Springer, Heidelberg (2009)
5. Yang, J.M., Cai, R., Wang, Y.D., Zhu, J., Zhang, L., Ma, W.Y.: Incorporating site-level knowledge to extract structured data from web forums. In: The 18th International Conference on World Wide Web, pp. 181–190. ACM, New York (2009)
6. Wang, H.N., Wang, C., Zhai, C.X., Han, J.W.: Learning Online Discussion Structures by Conditional Random Fields. In: Proceedings of the 34th Annual ACM SIGIR Conference (SIGIR 2011). ACM, New York (2011)
7. Xu, G., Ma, W.: Building implicit links from content for forum search. In: Proceedings of the 29th Annual ACM SIGIR Conference (SIGIR 2006), pp. 300–307. ACM, New York (2006)
8. Shi, X., Zhu, J., Cai, R., Zhang, L.: User grouping behavior in online forums. In: Proceedings of the 15th KDD, pp. 777–786. ACM, New York (2009)
9. Cong, G., Wang, L., Lin, C.Y., Song, Y.I., Sun, Y.H.: Finding Question-Answer Pairs from Online Forums. In: Proceedings of the 31th Annual ACM SIGIR Conference (SIGIR 2008). ACM, New York (2008)
10. Huang, J.Z., Zhou, M., Yang, D.: Extracting chatbot knowledge from online discussion forums. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI 2007). Morgan Kaufmann Publishers Inc., San Francisco (2006)
11. Wang, Y.D., Yang, J.M., Lai, W., Cai, R., Zhang, L.: Exploring Traversal Strategy for Web Forum Crawling. In: Proceedings of the 31th Annual ACM SIGIR Conference (SIGIR 2008). ACM, New York (2008)
12. Navarro, G.: A guided tour to approximate string matching. J. ACM Comput. Surv. 33(1), 31–38 (2001)
13. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. J. Data Min. Knowl. Disc. 2, 121–167 (1998)
14. Hu, F.H., Ruan, T., Shao, Z.Q., Ding, J.: Automatic Web Information Extraction Based on Rules. In: Bouguettaya, A., Hauswirth, M., Liu, L. (eds.) WISE 2011. LNCS, vol. 6997, pp. 265–272. Springer, Heidelberg (2011)

Characterizing Topic-Specific Hashtag Cascade in Twitter Based on Distributions of User Influence

Geerajit Rattanaritnont, Masashi Toyoda, and Masaru Kitsuregawa

The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{aomi,toyoda,kitsure}@tkl.iis.u-tokyo.ac.jp
http://www.u-tokyo.ac.jp

Abstract. As online social networks become extremely popular in these days, people communicate and exchange information for various purposes. In this paper, we investigate patterns of information diffusion and behaviors of participating users in Twitter, which would be useful to verify the effectiveness of marketing and publicity campaigns. We characterize Twitter hashtag cascades corresponding to different topics by exploiting distributions of user influence; cascade ratio and tweet ratio. The cascade ratio indicates an ability of users to spread information to their neighborhoods, and the tweet ratio measures how much each user participates in each topic. We examined these two measures on a real Twitter dataset and found three major diffusion patterns over four topics.

Keywords: Data mining, information diffusion, social network.

1 Introduction

In addition to real world communication, people can now keep in touch with each other on social networking sites such as Facebook, Twitter, and MySpace. People connecting to online social networks can share interests and activities with their friends, and even make new friends all over the world. The resulting networks grow rapidly and gained significant popularity on the Internet.

Many researchers have studied various aspects of online social networks such as network structure, user relationships, and information flow between users. In this paper, we perform a research on Twitter's user networks to understand patterns of information diffusion and behaviors of participating users. This would be useful to verify whether publicity campaigns are successful according to marketing strategies, such as aiming to spread mentions on new products to a large number of people or to a specific group of fans.

We analyze the diffusion of information according to topics of most frequently used hashtags and find the characteristics across them. To study a large amount of data, we consider two probability distributions of user influence: cascade ratio

and tweet ratio. The cascade ratio indicates an ability of a user to cascade information to his neighborhoods and the tweet ratio measures how much each user involves in each topic. We examine these two measures on a real Twitter dataset.

The Twitter dataset used in this paper is crawled from March 11, 2011 to July 11, 2011. It consists of 260 thousand users and 783 million tweets. We select top 100 frequently used hashtags from the dataset and categorize them according to topics. We found that the majority fall into four topics which are earthquake, politics, media, and entertainment. We then further investigate patterns of information diffusion by clustering hashtags based on distributions of those measures. Our results show that there are three patterns of hashtag cascade among four different topics.

The rest of this paper is organized as follows. Section 2 introduces related work on information diffusion. Section 3 explains the dataset. In Section 4, we describe two proposed distributions and investigate the characteristics of information diffusion over four major topics. Then we conduct further analysis by using clustering algorithm in Section 5. Finally, we conclude this paper and future work in Section 6.

2   Related Work

Information diffusion in online communities has been studied for a decade. Gruhl *et al.* [4] studied the dynamics of information propagation in weblogs. They investigated characteristics of long-running topics due to outside world events or within the community. Leskovec *et al.* [6] also studied information propagation in weblogs. They proposed a simple model that mimics the spread of information in blogspace and is similar to propagation found in real life.

Instead of blogsphere, Liben-Nowell *et al.* [7] traced the spread of information at individual level and found that information reach people in a narrow deep pattern, continuing for several hundred steps. Similarly, Sun *et al.* [10] conducted an analysis on information diffusion in Facebook and discovered that large cascade begins with a substantial number of users who initiate short chains.

In most recent years, as Twitter becomes one of the most popular micro-blogging services and allows us to obtain its data via Twitter API, it gains much interest from many researchers [2,3,5,8,9,11,12,13]. Romero *et al.* [9] studied information spread in Twitter and showed that controversial political topics are particularly persistent with repeated exposures comparing to other topics. Bakshy et al. [1] exploited information cascade to identify influencers in Twitter. Unlike others, we study characteristics of information diffusion over different topics in Twitter in term of cascade ratio and tweet ratio. Because we will understand how people interact with each other and how information is cascaded. Rather than identifying influencers as in [1,3,5,12], we can utilize this work to verify success of viral marketing strategy.

Topic	Examples	Total
Earthquake	jishin, genpatsu, prayforjapan, save_fukushima, save_miyagi	21
Politics	bahrain, iranelection, wiunion, teaparty, gaddafi	32
Media	nicovideo, nhk, news, fujitv, ntv	13
Entertainment	madoka_magica, akb48, atakowa, tigerbunny, anohana	11

3 Twitter Dataset

We crawled the Twitter dataset from Twitter API from March 11, 2011 when the Great East Japan Earthquake took place to July 11, 2011. Our data collection consists of user profiles, timestamp and tweet contents including retweets. We started crawling from famous Japanese users. We firstly got timelines of these users, then repeatedly expanded the set of users by tracing retweets and mentions in their timelines. We then obtained 260 million users as active users and 783 million tweets. Instead of friend-follower graph, we regard directed links among users when user A has at least one retweet from or mention to user B and call this relationship as outgoing neighborhood. We extracted 31 million links by considering only active users.

To study information cascade according to different topics, we treat a hashtag as a representative of the topic users talk about. We select top 100 frequently used hashtags from the dataset and manually categorize them into topics. We found that the majority belong to four major topics which are earthquake, politics, media, and entertainment. Table1 shows examples of hashtags in each topic. Earthquake topic is about the Great East Japan Earthquake, while politics topic is related to political issues and events all over the world especially the uprising events in the Middle East. Media topic is represented by communication channels, such as, television networks. Lastly, entertainment topic refers to television programs, movies and artists.

4 User Influence Distributions

4.1 Cascade Ratio

Cascade ratio determines the proportion of how much a user can influence his neighborhoods to spread a hashtag comparing to all users who used the same hashtag. We captured the cascade by tracing the time each user firstly used a given hashtag. Thus, cascade score of a user is the number of his immediate incoming neighborhoods that reposted the hashtag after him. The cascade ratio cr of a user u posting hashtag h is then defined as below:

$$cr(u,h) = \frac{C(u,h)}{U(h)} \ . \tag{1}$$

where $C(u,h)$ is the cascade score of the user u posting the hashtag h and $U(h)$ is a set of all users using h.
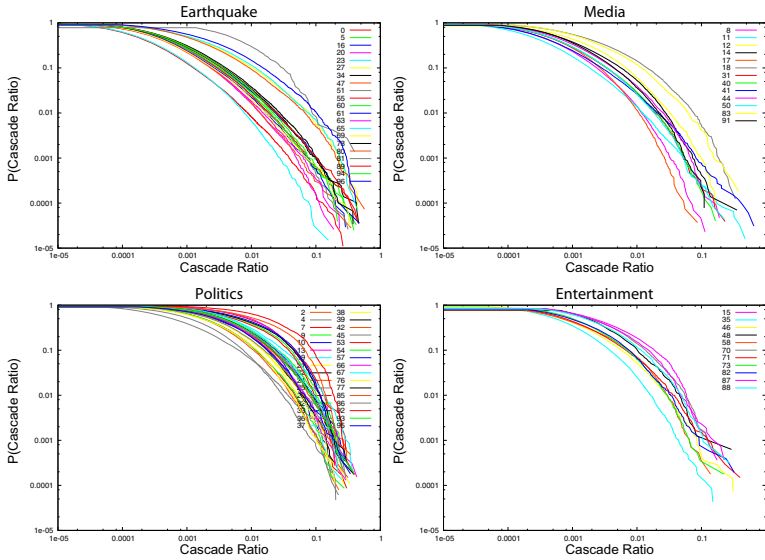
Fig. 1. Cascade ratio distributions of all hashtags in each topic

Fig.1 illustrates the probability distributions of cascade ratio of all hashtags according to four topics. The x-axis is cascade ratio and the y-axis is the number of occurrences of cascade ratios normalized by total number of users using a given hashtag. The plot is in log-log coordinate and calculated as a cumulative distribution function, where y or $P(x)$ is the probability at a value greater than or equal to x.

According to Fig.1, the earthquake and the media topics start to fall down at small cascade ratio. That means a number of people in these two topics have relatively low cascade ratio. It implies that people used those hashtags independently not because of seeing the hashtags in their friends' tweets. In other words, the hashtags themselves are hot topics or general words so that people know them already. For example, "jishin" in the earthquake category means earthquake in Japanese language and "nhk" in the media category is Japan's national public broadcasting organization. In contrast, the politics and the entertainment topics fall down at higher cascade ratio. We can say that a number of people have relatively high cascade ratio. When users post a hashtag in these topics, many of their friends will also post it after them. It means that there exist groups of users responding frequently each other, such as discussion communities on some political topics or fans of popular artists. For instance, "sgp" in the politics category is a non-profit organization for conservative women activists and "akb48" in the entertainment category is a popular Japanese female idol group.

For easy to see the difference among four topics, Fig.3a shows the point-wise average distributions of the cascade ratio for each topic. We see that 90% of all users who use hashtags in the earthquake and the media topics have cascade ratio

less than 0.005 which means they directly influence less than 0.5% of all users, while 2.7% and 0.8% for the politics and the entertainment topics respectively.

4.2   Tweet Ratio

The second measure is tweet ratio which determines how much each user engages in each topic. The tweet ratio tr of a user u posting hashtag h is then simply defined as below:

$$tr(u, h) = \frac{T(u, h)}{\sum_u T(u, h)} \ . \tag{2}$$

where $T(u, h)$ is the number of tweets containing the hashtag h posted by the user u.

Fig.2 shows the probability distribution of tweet ratio of all hashtags according to four topics. The x-axis is tweet ratio and the y-axis is the number of occurrences of tweet ratios normalized by total number of users using a given hashtag. Each line is plotted in log-log coordinate and calculated as a cumulative distribution function.
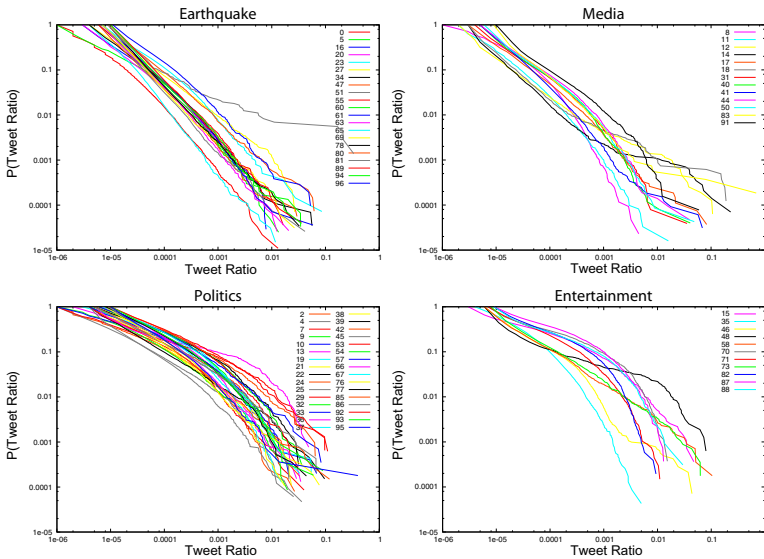
Fig. 2. Tweet ratio distributions of all hashtags in each topic

We see that the earthquake and the media topics follow the power-law, which means a number of people in the these topics have comparably low tweet ratio. They posted tweets containing hashtags but repeated to use those hashtags very few times. Alternatively, the politics and the entertainment topics are more curved. More people repetitively used same hashtags many times than people in the first two topics.

(a) Cascade ratio   (b) Tweet ratio
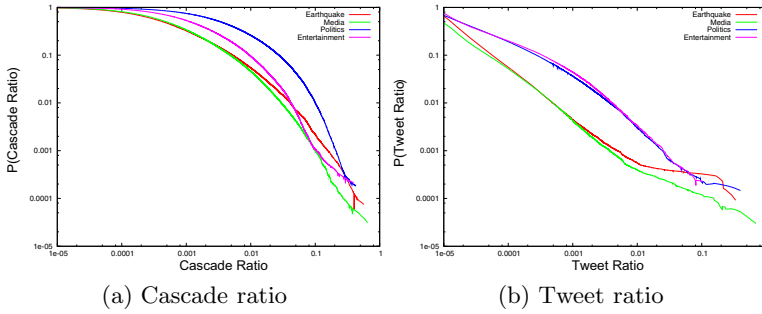
Fig. 3. Point-wise average distributions of four topics

Table 2. Proportion of each topic in each cluster when $k = 4$

No. of hashtags	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Total
Earthquake	1	13	4	3	21
Media	0	11	1	1	13
Politics	24	1	7	0	32
Entertainment	3	0	5	3	11

Additionally, Fig.3b demonstrates the point-wise average distributions of tweet ratio for each topic. We see that they are separate clearly into two groups. In more details, 90% of all users who use hashtags in the earthquake and the media topics have tweet ratio less than 0.00005, while 0.0003 in case of the politics and the entertainment topics.

5   Information Cascade Clustering

In this section, we further investigate the characteristics of information diffusion by using clustering algorithm. We performed k-means clustering based on the distributions of both cascade ratio and tweet ratio. Each hashtag is represented as a vector of values at n points in each distributions. For each hashtag, we selected 92 points proportional to the log scale from each distribution. We use Euclidean distance as a distance measure. Table2 illustrates the proportion of four topics assigned to each cluster when we choose the number of clusters as $k = 4$ as equal to the number of topics.

Fig.4 shows the point-wise average distributions of cascade ratio and tweet ratio according to four clusters. There are three main patterns among four clusters. Cluster 0 has high cascade ratio and high tweet ratio, cluster 1 has low cascade ratio and low tweet ratio, and cluster 2-3 are in the middle between them.

The majority of hashtags in cluster 0 are from the politics category, the majority of hashtags in cluster 1 are from the earthquake and the media categories, and the majority of hashtags in cluster 2 and 3 are from the entertainment category. However, we can see that cluster 2 includes various topics not only the

(a) Cascade ratio   (b) Tweet ratio
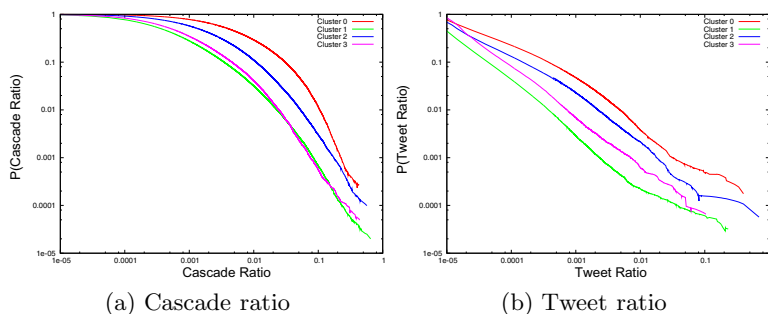
Fig. 4. Point-wise average distributions of four clusters

entertainment but also the earthquake, the media, and the political topics. For example, there are seven hashtags from the earthquake assigned into these clusters such as "iwakamiyasumi" and "hinan", which are related to nuclear power plants and safety information. They are thus considered to be used in longer period comparing to other earthquake hashtags in cluster 1. Moreover, there are seven hashtags from the politics category put into cluster 2. Almost of them are some of hashtags related to the Middle East such as "bahrain" and "egypt". Since they are country names, they are probably not limited only to political topic and thus more general than other political hashtags in cluster 0.

6 Conclusion

In this paper, we studied the characteristics of information diffusion according to topics of most frequently used hashtags in Twitter. Here we focused on four major topics which are earthquake, politics, media, and entertainment. According to the distributions of cascade ratio and tweet ratio, we have found that people in earthquake and media topics have less influence and use the same hashtags fewer times than people in politics and entertainment topics. Then, we also performed k-mean clustering based on both cascade ratio and tweet ratio. The results showed that we have three main patterns among four topics. The earthquake and the media topics have low cascade ratio and low tweet ratio, the politics topic has high cascade ratio and high tweet ratio, and the entertainment topic is in the middle among them. Finally, as future work, we need to explore other possible features, such as a number of followers and information cascade in term of geographical and temporal aspects, to characterize entertainment hashtags.

References

1. Bakshy, E., Hofman, J.M., Mason, W.A., Watts, D.J.: Everyone's an Influencer: Quantifying Influence on Twitter. In: 4th International Conference on Web Search and Data Mining, pp. 65–74. ACM (2011)
2. Castillo, C., Mendoza, M., Poblete, B.: Information Credibility on Twitter. In: 20th International Conference on World Wide Web, pp. 675–684. ACM (2011)

3. Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K.P.: Measuring User Influence in Twitter: The Million Follower Fallacy. In: 4th International Conference on Weblogs and Social Media, pp. 10–17. AAAI (2010)

4. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information Diffusion Through Blogspace. In: 13th International Conference on World Wide Web, pp. 491–501. ACM (2004)

5. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a Social Network or a News Media? In: 19th International Conference on World Wide Web, pp. 591–600. ACM (2010)

6. Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N., Hurst, M.: Patterns of Cascading Behavior in Large Blog Graphs. In: 7th SIAM International Conference on Data Mining, pp. 551–556. SIAM (2007)

7. Liben-Nowell, D., Kleinberg, J.: Tracing Information Flow on a Global Scale Using Internet Chain-Letter Data. The National Academy of Sciences, 4633–4638 (2008)

8. Meeder, B., Karrer, B., Sayedi, A., Ravi, R., Borgs, C., Chayes, J.: We Know Who You Followed Last Summer: Inferring Social Link Creation Times in Twitter. In: 20th International Conference on World Wide Web, pp. 517–526. ACM (2011)

9. Romero, D.M., Meeder, B., Kleinberg, J.: Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter. In: 20th International Conference on World Wide Web, pp. 695–704. ACM (2011)

10. Sun, E., Rosenn, I., Marlow, C., Lento, T.: Gesundheit! Modeling Contagion through Facebook News Feed. In: 3rd International Conference on Weblogs and Social Media, pp. 146–153. AAAI (2009)

11. Scellato, S., Mascolo, C., Musolesi, M., Crowcroft, J.: Track Globally, Deliver Locally: Improving Content Delivery Networks by Tracking Geographic Social Cascades. In: 20th International Conference on World Wide Web, pp. 457–466. ACM (2011)

12. Weng, J., Lim, E.-P., Jiang, J., He, Q.: TwitterRank: Finding Topic-Sensitive Influential Twitterers. In: 3rd International Conference on Web Search and Data Mining, pp. 261–270. ACM (2010)

13. Wu, S., Hofman, J.M., Mason, W.A., Watts, D.J.: Who Says What to Whom on Twitter. In: 20th International Conference on World Wide Web, pp. 705–714. ACM (2011)

A Study on Modeling of Lightweight Scientific Workflow Systems Using XML Schema⋆

Yingbo Liu[1,2], Feng Wang[1,2,⋆⋆], Hui Deng[1], Xiaodong Fu[1], and Kaifan Ji[1]

[1] Computer Technology Application Key Lab of Yunnan Province,
Kunming University of Science and Technology, Kunming, Yunnan, 650500, China
wangfeng@acm.org, {lyb,dh,jkf}@cnlab.net, xiaodong_fu@hotmail.com
[2] Yunnan Astronomical Observatory, Chinese Academy of Science,
Kunming, Yunnan, 650011, China

Abstract. Scientific workflow systems are typically describe coordinations involved in data processing via a workflow definition language. However, current specific workflow definition languages, even adopted by current mature scientific workflow systems, are too complex and enormous for non-professionals. In this paper, we mainly discuss how to design a workflow definition language using XML schema to suit a lightweight workflow system in a specific domain such as astronomy and astroinformatics. The prototype system shows that a simple but intuitive workflow definition language can simplify the modeling procedure and make end users easier and more accurate to implement their research tasks in e-Science activities.

Keywords: lightweight scientific workflow, workflow description language, modeling.

1 Introduction

As an important component of virtual astronomy technology, scientific workflow (SWF) is always the hot issue for research. Generally speaking, scientific workflow system is a kind of application software oriented in scientific experiments and workflow process. Although some mature scientific workflow systems, such as Kepler [10] and Taverna [9], with high practicability and abstractness, have been applied in many different domains. However, of wide application and various functions as they are, these systems are so enormous that they are usually confined to their own field. The astronomers always complain that these SWF systems are too complex and difficult to write programs. They do hope a simple, intuitive, easy and "lightweight" SWF system. A lightweight scientific workflow system is generally featured as follows: 1) Sufficient functions, 2) Not seeking for universality, and 3) Priority to data driven.

The first step to implement a scientific workflow is to consider how to describe a workflow, namely, process modeling of the workflow [17,1,15]. Its design

---

⋆ This work was funded by National Science Foundation of China (10878009).
⋆⋆ Corresponding author.

and implementation can be greatly influenced by the definition mode. This paper introduces in detail the language design in modeling process of lightweight workflow systems; how to define a simple and data driven workflow definition language with the existing XML technology, and the potential problems in process modeling of a scientific workflow system.

2   Workflow Structure Analysis

For an easy analysis of the modeling process, we firstly discuss an Integrated Abstract Single Workflow as an example (see Figure 1). Part U from the user's view [2] is the original and visualized illustration of a workflow. And Part C from the computer's perspective is an illustration to the actual detail of the internal workflow. This figure is a single workflow of single input and multiple outputs, representative for a data-driven workflow process. The detailed description is as follows:
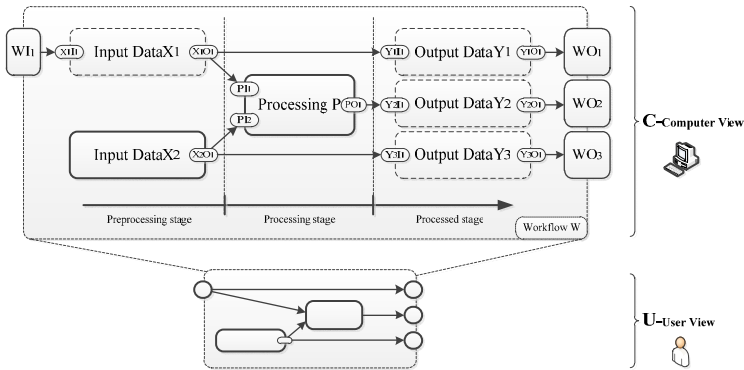
Fig. 1. Common Internal Structure Diagram for Single Workflow. The difference between user's and computer's comprehension to workflow lies in the offering abstract workflow to users while providing detailed and complex one to computer. To understand the internal structure of a workflow we need an analysis form the computer's perspective.

1) The whole workflow process is indicated as W;

2) The outermost dotted line box represents the application range of single workflow, indicated as W-boundary;

3) The workflow W has an input port WI_1, which is an interactive interface for communication with other workflows and application systems. It may complete data interaction according to their agreement. We can regard WI_1 as the abstraction of the data. In real system, these data may be a large text or a high resolution image.

4) The workflow W has output ports: WO_1, WO_2 and WO_3. These ports are the interactive interfaces between workflow W and other workflows or application

systems. The type and function of output data are similar to that of input port WI. Their difference lies in the data direction, as the input port can only be used for data input and the output port is just for data output, which cannot mix.

5) In workflow W, Input Data X_1 and Input Data X_2 are abstracted as data input channel in the workflow internal. They are used for data transmission or generating some process data. Input Data X_1 in Figure 1 represents data transmission or data buffering(via network to load distance data) without data processing; with data processing (such as data filter and correction, data generating and so on)they should be differentiated as Input Data X_2. As to Input Data X_1, it is a virtual data receiver with a single ended data input X_1I_1 and a single-ended output X_1O_1. And Input Data X_2 is a data generator with a zero end input and one end output (X_2O_1);

6) In internal workflow W, P is a processing unit, which can independently implement the executable unit of experiment process, representing data processing in practical application, such as the implementation algorithm processing for large data. In this study, the data process through processing unit is always indicated by solid lines, which means that it will implement data processing in all time. The input ports PI_1 and PI_2 receive original data, and data port PO_1 stores the data which processed by some sort of algorithm of PI_1 and PI_2. The output data lost the original data characteristics of PI_1 and PI_2, but its source information will be kept. For the need of data sourcing, workflow system will add some meta-information to the data processed by the execution unit.

7) In internal workflow W, Output Data Y_1, Output Data Y_2 and Output Data Y_3 in dotted line box respectively represent the transmission channel for the results from internal processing in workflow. They will not implement the data reprocessing, but play a role for data cache and sending. For example, if P is a scarce computing resource, it needs to release the occupied resources after data processing so that other applications can implement. However, if the receiving port WO_2 cannot obtain data in some conditions, it has to draw support from unit Output Data Y_2 for data cache. Such output channels as Output Data Y_1, Output Data Y_2 and Output Data Y_3 with cache function are always single end input and single end output.

3   Workflow Description Way Using XML Schema

The modeling method based on pipeline is fit for computer experts, but each processing link of the workflow described in this way is vague or even obscure. Against this background, the workflow description languages based on XML can effectively solve this problem.

Although workflow definition languages based on XML, such as XPDL, YAWL, WS-CDL and WSFL, seem to be huge to implement lightweight scientific workflow systems, we just need to adopt them or one of the subset to define new description languages. In this study, we discuss how to model a data-driven lightweight scientific workflow description language based on XML schema.

Through the analysis of Figure 1, we can abstract several key components as shown in Figure 2.
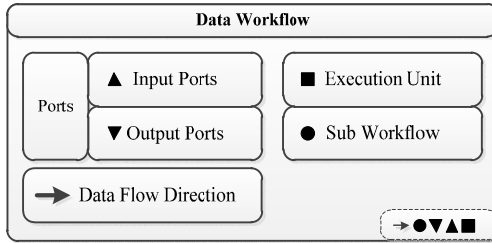
Fig. 2. Internal Workflow Units Constitutional Diagram after High Level Abstractness

In order to describe the example in Figure 1, it shall explain the foregoing units as Table 1.

Table 1. Keywords Definition of Workflow Description

Keywords	Specific Description
Ports	Define data input and output of workflow, and sub links Input Ports and Output Ports.
Execution Unit	Specific executive unit of workflow, named as Processor or Task in some famous workflow systems, which is common used. They are actually physically executable substance, which can be an application program or an executable service.
Data Flow Direction	Direction for data flow in different substances of the workflow.
Sub Workflow	Sub workflow, an unit consisting of commonly used functions, offering to others, which makes for function. [4]

To explain the relationship of units in Figure 2, we make a specific Scientific Process Definition Language (SPDL) to describe a workflow system. The definition can directly use the original way of XML, or on the basis of other languages based on XML, such as Scufl[9]. In this study the defined language is based on the original way of XML. Ports and links [13] involved in this definition is inspired by Taverna process description language — Scufl [9,14]. As to the workflow of part U in Figure 1, its description is as follows:

1) Define a top container to describe the workflow, named as Workflow:

```
<dataworkflow name='Workflow'>
```

2) Describe port: the workflow has 1 input port and 3 output ports. In data description, the copy process in the dotted lines units has no expenses cost, which can be omitted. Actually it is similar to the description process from the user's point of view, as shown in part U in Figure 1; Unit Input Data X_2 has one output port. Processing P have 2 input ports and 1 output port, mark for:

```
<ports>
    <inputports name=' Workflow:WI1'>
    <inputports name='ProcessingP:PX1'>
    <inputports name='ProcessingP:PX2'>
    <outputports name='InputDataX2:X201'>
    <outputports name=' ProcessingP:PO1'>
    <outputports name=' Workflow:WO1'>
    <outputports name=' Workflow:WO2'>
    <outputports name=' Workflow:WO3'>
</ports>
```

3) Describe processing units: as InputData X_2 and Processing both implement processing on data, it can be listed in Execution Unit:

```
<executionunits>
    <unit name='ProcessingP' oper='path to processing application'></unit>
    <unit name='InputDataX2' oper='path to data generator'></unit>
</executionunits>
```

4) Describe data flow direction: if we have data source and flow direction, together with data processing ways, now we can define a sub link $< links >$ to limit data flow direction:

```
<links>
    <link source=' Workflow:WI1' destination=' Workflow:WO1'>
    <link source=' Workflow:WI1' destination=' ProcessingP:PX1'>
    <link source=' InputDataX2:X201' destination='ProcessingP:PX1'>
    <link source=' ProcessingP:PO1' destination='Workflow:WO2'>
    <link source=' InputDataX2:X201' destination='Workflow:WO3'>
</links>
```

Through the above process it completes the description to Figure 1, which is the basic process of data-driven workflow system. According to the requirements of the description language for description language, it may add some other key words for description.

4   Analysis on Modeling Linking

As to the workflow description language based on XML, it shall pay much attention to some incorrect or nonstandard data performance listed in Figure 3 when modeling the workflow. The causes lie in the consistency treatment to workflow

port and executable unit port by the above description language, which make it easy to mix in the process of establishing description documents manually. So it should be noted that part or all of the following nonstandard workflow models shall result in incorrect execution of the workflow systems. As shown in Figure 3, some specific situations exist in the process of description based on XML:
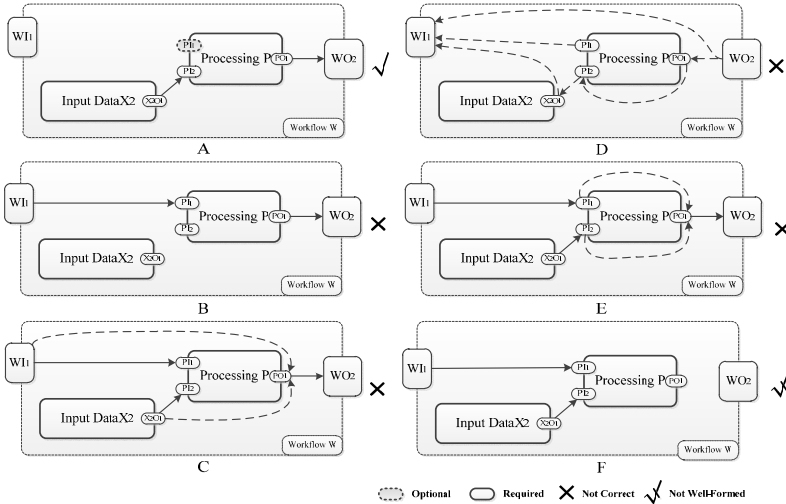
Fig. 3. It may have some specific modes when adopts workflow description language based on XML. In the figure, models except A are undemanding, and it is not allowed in actual description.

1) Any input port of executable unit shall be suspended, but the suspending port (such as PI_1 in A) is an optional port, and the data have default values. Although this model seems incorrect in its performance, it is actually an executable workflow.

2) As to PI_2 in B, the output data from this port is the required input. In this case, if there is no data input, the whole process cannot implement, so the model is regarded as the incorrect one.

3) In this model, it is not allowed to directly connect any input port to the output port of executable units.

4) In a single workflow, any input port (output port of executable unit or workflow) should not be directly connected to any internal input port which needs to process data. It is not allowed to connect input port to the output port.

5) In the executable units, it is not correct to directly establish data flow between input and output port.

6) Suspending output port of executable units does not accord with standard but can satisfy workflow execution.

When using workflow description language for workflow process modeling, three methods are exploited to avoid the above mentioned errors.

1) Design an assorted GUI graphical user interface to manage workflow description language. With tips on the interface the user may avoid nonstandard description. At present many workflow systems have their corresponding GUI management interface of workflow process description language, which is very convenient for operation to description language via GUI interface;

2) Design complete Schema specifications for description language. This method can verify effectively basic errors in workflow description;

3) Before implementing the workflow system, the engine shall check the integrity of the workflow's operation.

5   Related Work

There are many researchers studying modeling methods from the application level, but with various emphases. The most common modeling methods are based on Directed Acyclic Graphs (DAG) [7] and Directed Cyclic Graph (DCG) method[5,10].For the introduction to workflow description language, Tristan Glatard made a detailed analysis and classification [6]. Guan [7] discussed the workflow modeling and workflow language in detail and introduced modeling with Petri-Net. Maheshwari [11] discussed the workflow process description based on visualization and the method of script way; McPhillips concluded some matters needing attention for workflow design, and used an integrated modeling method [12]; Johan Montagnat adopts array programming way to analyze and design workflow, and discussed about collaboration of data flow and control flow and put forward some instructive suggestions. Gubala Tomasz [8] applied the abstract and concrete concept to describe workflow design, and Yang Ping [15] used hierarchical state machine (HSM) for workflow modeling.

In this paper, the mentioned models are different from the scientific workflow patterns [16] discussed by Ustun Yildiz. Models in this paper mainly focus on the performance mode of workflow modeling. As to the language definition it is similar to the data flow performance using Sucfl by Tom Oinn [13].

6   Conclusion

This study discusses the workflow structure in the range of single workflow and how to define a lightweight workflow description language on the basis of analysis. It analyzes in details the scientific workflow structure in single data flow, and deeply reveals the particulars of data flow. And it then makes detailed analysis on how to adopt XML to design a specific workflow system description language, discusses some common error modes of workflow definition, and put forward some suggestions. The situation discussed in this paper is in a single workflow case, which does not involve in the situation of more complex and multiple workflows, and this will be the research emphasis for the next step.

References

1. Adam, N.R., Atluri, V., Huang, W.K.: Modeling and analysis of workflows using petri nets. J. Intell. Inf. Syst. 10(2), 131–158 (1998)
2. Biton, O., Davidson, S.B., Khanna, S., Roy, S.: Optimizing user views for workflows (2009)
3. Bowers, S., Ludäscher, B.: Actor-Oriented Design of Scientific Workflows. In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, Ó. (eds.) ER 2005. LNCS, vol. 3716, pp. 369–384. Springer, Heidelberg (2005)
4. Bowers, S., Ludäscher, B.: Enabling scientific workflow reuse through structured composition of dataflow and control-flow (2006)
5. Churches, D., Gombas, G., Harrison, A., Maassen, J., Robinson, C., Shields, M., Taylor, I., Wang, I.: Programming scientific and distributed workflow with triana services: Research articles. Concurr. Comput.: Pract. Exper. 18(10), 1021–1037 (2006)
6. Glatard, T., Montagnat, J., Bolze, R., Desprez, F., Modalis, E.: On scientific workfkow representation languages. Tech. rep., Laboratoire d'Informatique Signaux et Systmes de Sophia-Antipolis (2009)
7. Guan, Z., Hernandez, F., Bangalore, P., Gray, J., Skjellum, A., Velusamy, V., Liu, Y.: Grid-flow: A grid-enabled scientific workflow system with a petri net-based interface. In: Grid Workflow Special Issue of Concurrency and Computation: Practice and Experience (2005)
8. Gubala, T., Harezlak, D., Bubak, M., Malawski, M.: Constructing abstract workflows of applications with workflow composition tool. In: Proceedings of Cracow 2006 Grid Workshop, pp. 25–30. ACC Cyfronet AGH (2006)
9. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M.R., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34(Web Server issue), W729–W732 (2006)
10. Ludascher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J., Zhao, Y.: Scientific workflow management and the kepler system: Research articles. Concurr. Comput.: Pract. Exper. 18(10), 1039–1065 (2006)
11. Maheshwari, K., Montagnat, J.: Scientific workflow development using both visual and script-based representation (2010)
12. McPhillips, T., Bowers, S., Zinn, D., Ludäscher, B.: Scientific workflow design for mere mortals. Future Gener. Comput. Syst. 25(5), 541–551 (2009)
13. Oinn, T., Addis, M., Ferris, J., Marvin, D., Greenwood, M., Goble, C., Wipat, A., Li, P., Carver, T.: Delivering web service coordination capability to users (2004)
14. Oinn, T., Greenwood, M., Addis, M., Alpdemir, M.N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M.R., Senger, M., Stevens, R., Wipat, A., Wroe, C.: Taverna: lessons in creating a workflow environment for the life sciences: Research articles. Concurr. Comput.: Pract. Exper. 18(10), 1067–1100 (2006)
15. Yang, P., Yang, Z., Lu, S.: Formal modeling and analysis of scientific workflows using hierarchical state machines (2007)
16. Yildiz, U., Guabtni, A., Ngu, A.H.H.: Towards scientific workflow patterns (2009)
17. Zinn, D.: Modeling and optimization of scientific workflows (2008)

Hit Count Reliability:
How Much Can We Trust Hit Counts?

Koh Satoh and Hayato Yamana

Fundamental Science and Engineering,
Waseda University, Bld.51-11-5, 3-4-1 Okubo Shinjuku-ku Tokyo, Japan
{kohsatoh,yamana}@yama.info.waseda.ac.jp

Abstract. Recently, there have been numerous studies that rely on the number of search results, i.e., hit count. However, hit counts returned by search engines can vary unnaturally when observed on different days, and may contain large errors that affect researches that depend on those results. Such errors can result in low precision of machine translation, incorrect extraction of synonyms and other problems. Thus, it is indispensable to evaluate and to improve the reliability of hit counts. There exist several researches to show the phenomenon; however, none of previous researches have made clear how much we can trust them. In this paper, we propose hit counts' reliability metrics to quantitatively evaluate hit counts' reliability to improve hit count selection. The evaluation results with Google show that our metrics successfully adopt reliable hit counts – 99.8% precision, and skip to adopt unreliable hit counts – 74.3% precision.

Keywords: Information Retrieval, Search Engine, Hit Count, Reliability.

1 Introduction

Recently, there exist many studies that rely upon search engines' hit counts—the estimated numbers of result pages [1 - 5]. Since these studies are conducted under the supposition that a hit count can be an indicator of term frequency for a query in the whole Web, the reliable hit counts have become indispensable.

Hit counts are widely used especially in the field of natural language processing researches [1 - 3]. In addition, more recently, hit counts are also used in other fields, such as ontology construction [4] and automatic extraction of social-networks [5], and are becoming increasingly important as a research tool.

However, hit counts returned by search engines are known to be unreliable [6 - 9]. In our previous research, we found that hit counts for a query sometimes changed by more than 10 to 100 fold over a short period of just one or two days. As is shown by this example, hit counts can cause too large errors to be used for studies.

The main purpose of our work is to avoid adopting these unnatural hit counts, and instead to choose only reliable hit counts, i.e. not to choose the hit counts when they are unnaturally changing. In this paper, we propose hit counts' reliability metrics that

enables us to evaluate hit count reliability quantitatively by using each search engine's hit counts transition. We assume that drastically changed or bursting hit counts have low reliability because of search engines' index updating or index inconsistency. Then, we also propose a method to choose reliable hit counts.

2   Related Work

Related researches are classified into two groups; researches utilizing search engines' hit counts and researches verifying hit counts' reliability.

2.1   Studies Utilizing Hit Counts

Turney [3] proposed a synonym extraction method called "PMI-IR" using search engine's hit counts to identify the word with the most similar meaning to the target word among several alternatives, which is commonly seen in TOEFL questions. In this method, scores are calculated for each alternative as below.

$$Score(choice_i) = hits(problem\ AND\ choice_i)\ /\ hits(choice_i) \qquad (2.1)$$

where, *problem* is the target word, *choice_i* is the *i*-th alternative word and hits(Q) is the hit count obtained when querying a search engine for a word Q. In this method, we see that if the comparative ordering of the hit counts for alternatives happens to turn over because of hit count fluctuation, then the extracted synonym will change.

Cilibrasi et al. [2] proposed a word similarity measurement using hit counts called "Google Similarity Distance." It utilizes the hit counts of an "AND search" to obtain the degree of co-occurrence of two words, and defines the similarity between two queries.

As is shown by these examples, hit counts are widely used by natural language processing researchers who regard the whole Web as a corpus. When using hit counts, many studies adopt hit counts in order to determine which word, or phrase, among alternative queries is more commonly used on the Web. Thus, a correct comparative ordering of queries' hit counts is indispensable.

2.2   Studies on Hit Counts' Reliability

Thelwall [6][7] compared the hit counts of three search engines - Google, Yahoo! and Live Search. He selected 2,000 words that had various hit counts, and calculated the correlation of hit counts between search engines. His research focused on discussing the differences in the hit counts from several search engines, but did not concern itself about evaluating hit counts' reliability, nor how to select reliable hit counts.

Uyar [8] investigated the accuracy of hit counts returned by Google, Yahoo! and Live Search. He assumed that the correct number of the hit count is the number of Web pages actually obtained only when the number of obtained search results is less than 1,000, the maximum number of results each search engine provides for a given query. Based on the assumption, he investigated the accuracy of hit counts for 1,000

queries by calculating "*Percentage of Error*" only when the hit count is less than 1,000 as below.

$$(\#ofEstimate - \#ofReturnedDocument) \: / \: \#ofReturnedDocument \times 100 \qquad (2.2)$$

His main goal was to compare three search engines to find out the search engine returning the most accurate hit counts, but he did not investigate the hit count transition during days within a search engine.

In our previous work, Funahashi et al. [9] gathered hit counts of 10,000 queries for three search engines - Google, Yahoo! and Bing - and investigated the transition tendencies and discussed the conditions where search engines return reliable hit counts. They concluded that the most reliable hit counts are ones with the largest search offset just before the search engines adjust their hit counts to the actual number of returned documents when they are consistent, i.e., less than 30% change over approximately a week. However, their work is insufficient by not clearly defining a "reliable hit count," and did not evaluate the propriety of the proposed method.

In this paper, we enhance our previous research by defining the metrics to evaluate hit counts' reliability, and propose the method to choose reliable hit counts.

3   Hit Count Transition Patterns

In this section, we inspect the transition tendencies which can be seen when observing hit counts for a long term, and then discuss what causes hit count change.

Fig. 1 shows the transition of Google's hit counts for the query "canon" from July 2010 to Nov. 2011. The typical transition patterns of hit count are classified into the following four patterns as shown in Fig. 1.

I.   Stable part with less change, which occupies most of the observation term
II.   Drastically changing part over several days
III. A one-day spike
IV.   A continuous spike over a few days

Phase I, most of the observation term, shows smooth change when search engines gradually increment their indexes. On the contrary, phase II, III and IV show drastic change of hit counts. We assume these phases are resulted from 1) update of large part of index, 2) inconsistency of indexes, and 3) connecting a different search unit. The followings are discussion on the causes of hit count's change.

Index Update. Search engines are supposed to crawl at short intervals and update their indexes. [12] describes that search engines have small parts of index updated in seconds as well as large ones updated over days or weeks. The update of the small parts of index may cause the transition pattern of phase I and the update of the large parts may cause that of phase II.

Inconsistency between Multiple Indexes. Search engines usually try to reduce the number of accesses to full index by having a cache structure [13]. When inconsistencies between result-cache and full-index exist, different hit counts can be observed depending on the indexes, which can be explained as the phenomena in phase III.

Connecting to a Different Search Unit. Most of major search engines allocate their datacenters all over the world that are comprised of many servers with indexes, in order to serve numerous queries from all over the world [14]. The indexes of these search units are supposed to be synchronized with one another, but may have inconsistencies, especially during periods of index-updating, which can be explained as the phenomena in phase III and phase IV.

The main purpose of this work is to statistically evaluate the indispensable number of observation days required to guarantee that the obtained hit count is in phase I with a given probability based on a large-scale hit count observation dataset.
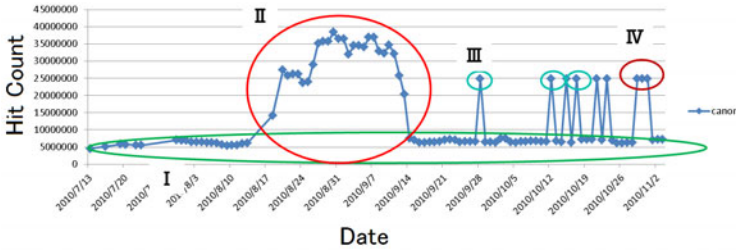
Fig. 1. Google's hit count transition for query "canon"

4   Reliability Evaluation Metrics

As discussed in Sec. 2.1, most studies using hit counts utilize their comparative ordering instead of their absolute values. This means that turnover in the comparative ordering of hit counts will have adverse effects on such studies. Thus, we assume that a pair of hit counts whose comparative ordering turns over frequently, where in phase II - IV, is unreliable. Instead, we assume that a pair of hit counts in phase I is reliable.

In this paper, we propose reliability evaluation metrics for hit counts to guarantee that the obtained hit counts are in phase I, i.e. stable phase.

4.1   Definition of Metrics

We propose a reliability function of a pair of hit counts' relation. Here, *reliability(hit[A]>hit[B], m)* represents the probability that the two hit counts *hit[A]* and *hit[B]* keep the same relationship of *hit[A]>hit[B]* for *m*-days after the hit counts for query *A* and *B* are initially observed, as shown in (4.1).

$$reliability\big(hit[A] > hit[B], m\big)$$
$$= \Pr\big(days(hit[A] > hit[B]) > m\big) \tag{4.1}$$

where, *days(hit[A]>hit[B])* represents the number of consecutive days for which a certain comparative ordering of queries (*hit[A]>hit[B]*) is unchanged.

4.2   Reliability Evaluation Experiment

Since this reliability will depend on the number of days to observe that the comparative ordering between the queries is unchanged, we performed an experiment to

calculate a *Reliability Function* below by using a large-scale hit count's observation dataset to confirm the relation.

$$reliability \ function \ :< R, a, m > \rightarrow r$$

$$where \quad r = reliability \ y \begin{pmatrix} hit[A] > hit[B], m \\ | \ hit_{base}[A] : hit_{base}[B] = R : 1, \\ a = observedda \ ys(hit[A] > hit[B]) \end{pmatrix} \qquad (4.2)$$

Here, R represents the hit count ratio between query A and query B, i.e. *hit$_{base}$[A]/hit$_{base}$[B]*, on the first day of observation (*base*), a-days represents the number of days to observe that the comparative ordering of hit counts does not turn over, m-days represents the number of days for which a user wants to guarantee the same comparative ordering of two queries, i.e. unchanged during m-days.

Finally, r represents the reliability, i.e., the probability that the hit counts of two queries maintain the same comparative ordering for m-days. In other words, this function indicates the reliability under the conditions that the ratio of two queries' hit counts on the first *base* is R:1 and the comparative ordering is stable for consecutive a consecutive days.

4.3   Dataset and Results

We used 10,000 queries provided by Yahoo! JAPAN as the top 10,000 frequent queries in December 2007, because they included popular words. We observed hit counts of those queries for three search engines; Google, Yahoo! and Bing from Oct. 2009 to Dec. 2010, by using APIs each engines provides.

Ordering Guaranteed Days v.s. Reliability. Fig. 2 (a) shows the relation between m (number of days required to guarantee the comparative ordering) and reliability r with Google's hit counts when R is set to 2.0. This graph indicates that as a (number of days to observe the comparative ordering is stable) increases reliability also increases. This means that if we have a longer observation period, i.e. large a, we can have more reliable comparative ordering of hit counts.

Hit Counts' Ratio v.s. Observation Days. Fig. 2 (b) shows the relation between R (hit counts' ratio between two queries) and a (number of observation days) with
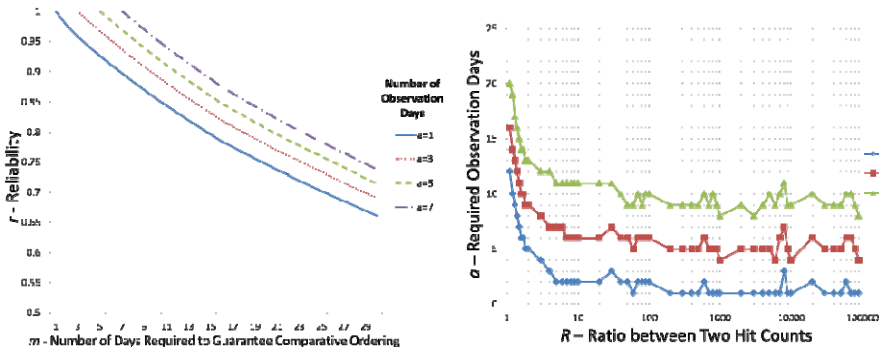
(a) *m-r* graph (R = 2.0)   (b) *R-a* graph ($r \geq 0.85$)

Fig. 2. Google's *m-r* graph and *R-a* graph

Google's hit counts under the condition that reliability r is over 85%. By using this graph, we are able to set the indispensable observation days to guarantee the comparative ordering of hit counts. For example, we see from the graph that if we desire 85% reliability that the comparative ordering of a pair of hit counts stay unchanged for 15 days where the initial hit counts' difference $R = 10$, then we must observe for two days that the ordering is unchanged.

5   Selection of Reliable Hit Counts

In this section, we propose a method to select reliable hit counts using the reliability evaluation metrics defined in the previous section.

5.1   Method of Selecting Reliable Hit Counts

Based on the *reliability function $<R, m, a> \rightarrow r$* described in Sec.4.2, we propose a method to select reliable hit counts. The method uses both the reliability r of each hit count and a user defined m-days where to guarantee the comparative ordering.

step1. Gather hit counts for the two queries, then calculate the ratio R between the two.
step2. Because we've already observed the hit counts for one day, let $a = 1$.
step3. Use the *reliability function* for the reliability r which corresponds to the triple $<R, m, a>$.
step4. If the obtained reliability r is high enough, then adopt the ordering and finish.
step5. If obtained reliability r is not high enough, check if the comparative ordering is unchanged on the next day. Then, let $a = a + 1$, and go back to step2.
step6. If the comparative ordering of the hit counts between two queries turns over, then go back to step1.

5.2   Experiment

Preparation. We gathered the hit counts data for 10,000 queries from Sep. 2010 to Jan. 2011. First, we divide the data into two at a date D; reliability-function-calculating-period just before date D and method-verification-period just after date D. Here, D is varied from Oct. 2010 to Dec. 2010.

Then, we calculate the *Reliability Function* with the data during T-days just before date D. Then, we evaluate the method for the hit counts during m-days after date D.

As for the correct comparative orderings, we adopt the ordering which is more observed in method-verification-period as shown in Table 1. Though we do not know the correct orderings as described in Sec.3, we have decided to adopt the more frequently observed ordering in method-verification-period as the correct orderings. In Table 1, *count(hit[A]>hit[B])* represents the number of days when hit counts of query A are over those of query B during m-day just after date D, and θ represents the threshold of deviation between *count(hit[A]<hit[B])* and *count(hit[A]>hit[B])* used to judge which ordering is correct.

Metrics – Error Ratio and Skip Ratio. *Error Ratio* and *Skip Ratio* shown in Fig. 3 are used to evaluate our proposed method. *Error Ratio* represents the ratio, how *incorrect orderings* are unsuccessfully adopted, compared to the total number of

Table 1. Definition of *Correct* and *Incorrect Ordering*

If (count(hit[*A*]>hit[*B*])/*m* > *θ*)
   then *correct_ordering* = hit[*A*]>hit[*B*]
Else If (count(hit[*A*]<hit[*B*])/*m* > *θ*)
   then *correct_ordering* = hit[*A*]<hit[*B*]
Else
  correct_ordering = UNDEFINED
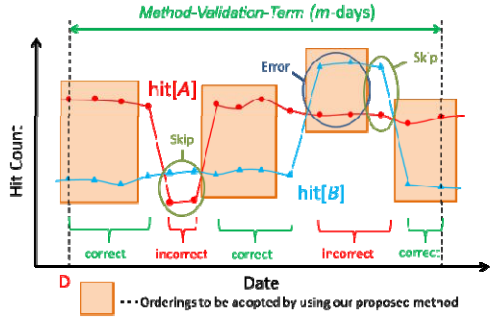
Fig. 3. Correct and *Incorrect*, *Error* and *Skip*

adopted orderings of hit counts (***number of errors/number of adoptions***). *Skip Ratio* represents the ratio, how many *incorrect orderings* are successfully skipped compared to the total number of incorrect orderings (***skips/number of incorrect orderings***).

In our experiment, we counted *number of errors*, *number of adoptions*, *number of skips* and *number of incorrect orderings* among 1,000,000 combinations of randomly chosen two queries selected from original 10,000 queries.

5.3   Results of Validation Experiments

Validation of Reliability Evaluation Metrics. Fig. 4 shows the Google's transition of *Error Ratio* and *Skip Ratio* from Oct. 2010 to Dec 2010, where $m = 20$ and $θ = 0.85$.

Here, *baseline* of each graph represents *Error Ratio* or *Skip Ratio* when adopting all comparative orderings of hit counts as it is without any filtering. By applying the proposed method we are able to avoid adopting the incorrect comparative orderings, which results in decreasing *Error Ratio* and increasing *Skip Ratio*.

As shown in Fig. 4, when we set reliability threshold $r > 0.9$, *Error Ratio* maintains a low value throughout the experimental period (1.5% at most), and *Skip Ratio* keeps a high value except for some parts of the experimental period. For Google, when we set reliability threshold $r = 0.9$, the total average of *Error Ratio* is 99.8% and the total average of *Skip Ratio* is 74.3%. Both of these results strongly support the validity of the hit count reliability evaluation metrics shown in Sec. 4.1.
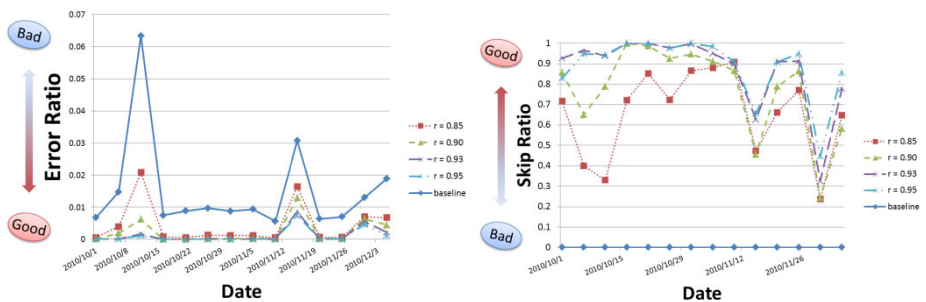
Fig. 4. Google's *Error Ratio* and *Skip Ratio* ($m = 20$, $θ = 0.85$)

6   Conclusion

In this paper, we defined the metrics for evaluating the reliability of hit counts, and proposed a method to select reliable hit counts based on hit count observation data.

We evaluated the validity of our method using a hit count observation dataset of 10,000 queries for Google, to confirm whether we can avoid using unstable comparative ordering of hit counts. The results show that our metrics successfully adopt reliable hit counts and skip to adopt unreliable hit counts.

Our future work will evaluate the proposed method more accurately by using static word-count data of the Web, not estimated number. Besides that, we are planning to distribute the reliability function data to researchers.

References

1. Grefenstette, G.: The World Wide Web as a resource for example-based machine translation tasks. In: Proc. of the ASLIB Translating and the Computer Conf., London (1999)
2. Cilibrasi, R., Vitanyi, P.M.B.: The Google similarity distance. IEEE Trans. on Knowledge and Data Engineering 19(3), 370–383 (2007)
3. Turney, P.: Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 491–502. Springer, Heidelberg (2001)
4. Cimiano, P., Handschuh, S.: Towards the self-annotating web. In: Proc. of 13th Int'l World Wide Web Conf. (WWW 2004), pp. 462–471 (2004)
5. Matsuo, Y., Mori, J., Hamasaki, M., Ishida, K., Nishimura, T., Takeda, H., Hashida, K.: POLYPHONET: An advanced social network extraction system. In: Proc. of 15th Int'l World Wide Web Conf. (2006)
6. Thelwall, M.: Extracting accurate and complete results from search engines: Case study windows live. J. of the American Society for Information Science and Technology 59(1), 38–50 (2007)
7. Thelwall, M.: Quantitative comparisons of search engine results. J. of the American Society for Information Science and Technology 59(11), 1702–1710 (2008)
8. Uyar, A.: Investigation of the accuracy of search engine hit counts. J. of Information Science 35(4), 469–480 (2009)
9. Funahashi, T., Yamana, H.: Reliability Verification of Search Engines' Hit Counts: How to Select a Reliable Hit Count for a Query. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 114–125. Springer, Heidelberg (2010)
10. Huang, Z., van Harmelen, F.: Using Semantic Distances for Reasoning with Inconsistent Ontologies. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 178–194. Springer, Heidelberg (2008)
11. Ping, I.C., Shi-Jen, L.: Automatic keyword prediction using Google similarity distance. Expert Systems with Applications 37, 1928–1938 (2010)
12. Challenges in Building Large-Scale Information Retrieval Systems, http://research.google.com/people/jeff/WSDM09-keynote.pdf
13. Skobelsyn, G., Junqueira, F., Plachouras, V., Baeza-Yates, R.: ResIn: A combination of results caching and index pruning for high-performance Web search engines. In: Proc. of 31st SIGIR, pp. 131–138 (2008)
14. Barroso, B.A., Dean, J., Holzle, U.: Web search for a planet: The google cluster architecture. IEEE Micro 22(2), 22–28 (2003)

A Topological Description Language for Agent Networks

Hao Lan Zhang[1], Chaoyi Pang[2], Xingsen Li[1], Bin Shen[1], and Yan Jiang[3]

[1] NIT, Zhejiang University, 1 Xuefu Road, Ningbo, China
[2] The Australian E-health Research Centre, CSIRO, Australia
[3] University of Southampton, UK
{haolan.zhang,lixs}@nit.zju.edu.cn, Chaoyi.Pang@csiro.au,
tsingbin@zju.edu.cn, yj6g10@ecs.soton.ac.uk

Abstract. Existing multi-agent systems lack a comprehensive methodology for agents to perceive the structures of a network, which they are residing in. This problem obstructs efficient agent communication and transmission within a multi-agent system. The demand for an efficient mechanism allowing an agent to sense other agents' locations has been growing substantially, since agents are mobile and cooperative entities. In this paper, we propose a new Topological Description Language for Agent networks, namely TDLA, to provide agents with the capability of understanding their environment. TDLA helps an agent perceive the network structure that it resides in. It adopts traditional topological analysis methods for local area networks and further tailored for agent networks.

Keywords: Agent Topological Language, Agent Network Topology, and Multi-agent Systems.

1 Introduction

Mathematical topology primarily deals with such fundamental notions as connectivity, transformations, invariance, the properties of surfaces, and the nature of space itself [1]. Nowadays, the topological concept has been extensively applied to various sub-domains of computer science, such as communication network constructions, multiprocessor systems, image processing, etc. However, the application of topological methodologies to multi-agent systems is still preliminary.

Agent Oriented Software Engineering (AOSE) is a promising field. Unlike many traditional software design methodologies, which mainly focus on the overall structural and procedural design for an entire system, AOSE design methodologies emphasise on the intelligent issues of individual components and granting autonomy to basic system units (i.e. agents). Each AOSE-based agent has the capability to perceive the environment and perform tasks intelligently and cooperatively [2, 3, 4]. Therefore, it is crucial for agents to comprehend other agents' information including location, functionality, etc. However, existing technology for agent-based systems is insufficient to provide

efficient cooperation mechanisms due to the lack of a systematic topological theory for agent networks.

In this paper we further extend the previous topological theory for multi-agent networks [5, 6]; and provide a set of tentative mechanisms employed in TDLA, which can improve agent transmission and communication efficiency.

2   Related Work

Local Area Networks (LANs) utilises topological descriptions to describe the layout of networks. LAN designers and administrators can easily understand the physical layout of a LAN through its topological description [7, 8]. However, existing LAN topologies adopt graph-based descriptions, which mainly reply on human-participated perception. The topological descriptions for LANs are insufficient to provide information for complex network operations without human intervention. Therefore, the topological descriptions for LANs are not suitable for networks that involve intelligent network operations, such as intelligent searching or matching. A multi-agent-based network is a typical network that cannot be simply described by the LAN-based topologies. In the circumstances, more comprehensive topological methodologies, such as topological description languages, are required to fulfil the needs of agent-based networks.

Previously, a topology description language, i.e. Candela, has been suggested to describe topologies of distributed-memory multiprocessor systems [9]. Candela makes use of some operation methods from programming languages such as loops, recursion, replication, etc. Recent research work in this field has paid more attention to semantic network description method, such as [10, 11]. The concept of Network Description Language (NDL) suggests using semantic description methods, such as RDF and semantic web, to describe network topology [11, 12, 13]. The semantic-based network description methods throw light on developing an intelligent and intuitive mechanism for agent network discovery; and they are particularly useful for mobile agents and agent matching processes. Therefore, the authors suggest a new language for describing agent network topologies, i.e. TDLA [6], which adopts some concepts from Candela for operations among various sub-networks. Nevertheless, Candela is essentially a graph description language. TDLA further extends Candela through enabling agents to understand the network structures that they reside in through incorporating graph-based description and semantic expression.

3   Topological Theory for Agent Networks

Agent Network Topologies (ANTs) have been systematically classified into two major categories, i.e. simple ANTs and complex ANTs [5]. TDLA is based on this ANT classification method.

3.1   Fundamental Structure of TDLA

In this section we discuss the fundamental structure of TDLA. Topological Description Language for Agent networks (TDLA). The detailed information of the TDLA is described in [6].

In a TDLA-based multi-agent system, each agent carries a set of identifiable TDLA descriptions. The TDLA descriptions contain information including individual agent description, (IAD), main agent-groups description (MAD), and overall agent-network description (OAD) [6]. These descriptions are stored in the Travel Control Centre (TCC) component. Each travelling agent needs to carry these descriptions before travelling. In a TCC, an Itinerary set component contains the information about a travelling agent's routes. A travelling agent may transit at several temporary sites before reaching to its destination; the Itinerary set contains the information on all the routes and each route is stored in an itinerary. The choice of the travelling route is based on various factors, which include the network traffic factor, the network topology factor, and the connectivity between origin and destination. The IAD, MAD, and OAD determine the network topology factor. The Communication and Language Centre (CLC) is the source of providing travel information. The BDI model also participates in TCC; it often determines whether an agent needs to travel based on the BDI reasoning results. In some cases, agents can determine whether they need to travel based on the information from the CLC.

The first step in the travelling process is obtaining and decomposing the travel information. The original travel information consists of a destination site name (if it is unique) or a specific address, the travel mode (one-way, round trip), message content, and communication language.

The second step of the travelling process is applying the IAD, MAD, and OAD to the itinerary selection process and generating the travel routes for the itinerary set.

The final step of the travelling process is: an agent travels according to the itinerary set, which is based on the First-In-First-Out (FIFO) principle. In other words, if there are several routes in the itinerary set, the agent will travel based on the first route in the set. When the agent enters the destination site then the first route in the set will be removed and the second first itinerary becomes the first route, as shown in Figure 1. To enter into the destination site the agent needs to have a ticket, so the agent sends an entry request to the host site and it can enter the host site once it gets an entry ticket, then the first itinerary is complete. If the itinerary set is empty then the agent has reached the final destination.

| Route Completion | ← | Route 1 | Route 2 | …… | Route N | ← | IAD + MAD + OAD |
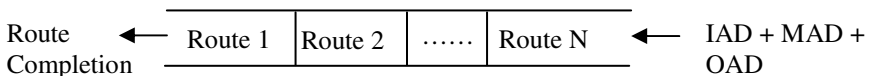
Fig. 1. FIFO Principle in Itinerary Set in TCC

3.2   Mechanisms for Determining Simple ANTs

In this paper we further formulise simple ANTs to support TDLA. Agents can use these formulised mechanisms to partially perceive the network structure they are

residing in. An agent can perceive a part of a complex or large network since it can only store a certain portion of an overall network information. Exceptions might occur when related topological information can be hold by an agent. The formulations described in this section adopt the following assumptions and notations:

- $f(x, y)$ is defined as a connection between two agents (nodes) x and y. If there is a link between x and y then $f(x, y) = 1$, if there is no link between x and y then $f(x, y) = 0$. In this paper there is only one link between any two nodes, therefore the dual-ring or dual-linear topologies are all regarded as single-line topologies.
- $\langle S, \tau \rangle$ is an agent network topological space, where S is a nonempty set that includes all the agents in the topological space and τ is the collection of all subsets of S.
- The total number of agents (nodes) in a network is N.

1) Constraints for centralised ANTs

$\exists A \in S$; $\exists P_i, P_j \in \langle S - A \rangle$. If S satisfies all the following constraints then S is a centralised ANT and A is the central agent:

- $\exists f(A, P_i) := 1$
- $\forall (P_i, P_j) \rightarrow \sum f(P_i, P_j) = 0$ (P_i, P_j denote all the other agents in the network except A).

- $\forall (P_i, P_j, A) \rightarrow \sum f(P_i, A) + \sum f(P_i, P_j) = N - 1$
- The following figure shows a typical centralised ANT.
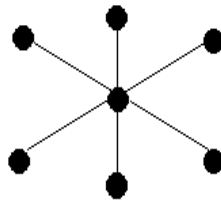
Fig. 2. Typical centralised ANT

2) Constraints for decentralised ANTs

Current definitions for decentralised networks (sometimes also known as peer-to-peer networks) are generally the opposite meaning of centralised networks, which allow nodes to communicate and share computational tasks and resources without central controls [14]. Decentralised networks are often classified into two major categories including partially connected decentralised networks and fully connected decentralised networks. In real world, partially connected decentralised networks lie somewhere between fully connected decentralised networks and closed-loop networks. Therefore, we have the following constraints for decentralised ANTs:

- $\exists P_i, P_j \in < S \text{:} >$
- $\forall(P_i, P_j) \rightarrow f(P_i, P_j) \geq 1$ (each agent in S is connected);
- $\forall(P_i, P_j) \rightarrow N < \sum f(P_i, P_j) \leq \dfrac{N(N-1)}{2}$ (The ANT is a fully connected

  decentralised ANT when the total number of connections equals to $\dfrac{N(N-1)}{2}$).

The following figure shows a typical peer-to-peer ANT.
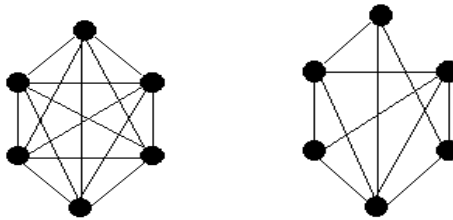
Fig. 3. Typical peer-to-peer ANTs

3) Constraints for broadcasting ANTs

$\exists P_i, P_j \in S$; $B \notin S$ and B is a common media. If S satisfies all the following constraints then S is broadcasting ANT:

- $\forall(P_i, P_j) \rightarrow f(P_i, P_j) = 0$;
- $\forall P_i \rightarrow f(P_i, B) = 1$.

The following figure shows a typical broadcasting ANT.
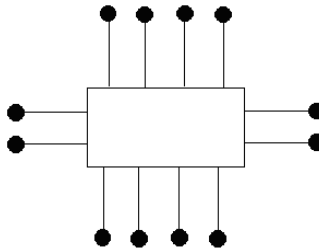
Fig. 4. Typical broadcasting ANT

4) Constraints for closed-loop ANTs

$P_i, P_j \in S$. If S satisfies all the following constraints then S is a closed-loop ANT:

- $\exists f(P_i, P_j) = 1$;

- $\forall(P_i, P_j) \rightarrow \sum f(P_i, P_j) = N$ (P_i is all the agents in S, P_j denotes all the other agents expect current P_i);

- For each $P_i \rightarrow \sum f(P_i, P_j) = 2$.

The following figure shows a typical closed-loop ANT.
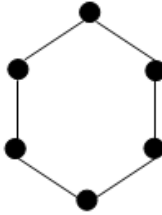
Fig. 5. Typical closed-loop ANT

5) Constraints for linear ANTs

$\exists P_1, P_2, P_i, P_j \in S$. If S satisfies all the following constraints then S is a linear ANT. $\sum f(P_1, P_i) = 1$ and $\sum f(P_2, P_i) = 1$ (P_1, P_2 are two end agents in the network):

- $\forall(P_i, P_j) \rightarrow \sum f(P_i, P_j) = N - 1$ (P_i, P_j are all the agents in S);
- For each $P \in \langle S - P_1 - P_2 \rangle \rightarrow \sum f(P, P_i) = 2$.

The following figure shows two typical linear ANTs.
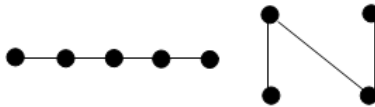
Fig. 6. Typical linear ANT

6) Constraints for hierarchical ANTs

$\exists P_i, P_{j1}, P_{j2}, P_{jn}, P_1 \in S$;

- P_1 denotes the root of the network;
- $\forall P_i \rightarrow \sum f(P_1, P_i) \geq 1$ (at least one child is connected to the root);

- $\forall (P_i, P_j) \rightarrow \sum f(P_i, P_j) = N - 1$;

- $f(P_i, P_{j1}) = 1 \;\wedge f(P_{j1}, P_{j2}) = 1 \;\wedge (P_{j2}, P_{jn}) = \; 1 \rightarrow f(P_i, P_{jn}) = 0$

To distinguish hierarchical ANTs from linear ANTs and centralised ANTs is that: a hierarchical ANT has a nominated root. The following figure shows a typical hierarchical ANT.
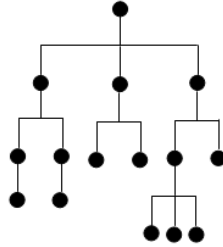
Fig. 7. Typical hierarchical ANT

4   Conclusion

This paper introduces a novel agent communication language and the topological description language for agent networks, i.e. TDLA, which adopts semantic-based method for describing agent network topologies. TDLA can be used for agent transmission and automated agent discovery processes. It overcomes the limitations of graph-based topological description methods and improves agents' capability of perceiving network topologies.

The TDLA method improves the efficiency of agent transition and discovery in a large agent network. The mechanisms for determining simple ANTs suggested in this paper are essential for agents to perceive the network information. Agents can apply their perceived network information and determine an optimised path for transition and communication. TDLA can be further extended to support complex ANTs, such as Small-world ANT [15, 16] or Scale-free ANT [17].

References

1. Wilton, W.B.: A – Z of Business Mathematics, p. 136. William Heinemann Publication (1980)
2. Bresciani, P., Giorgini, P., Hiunchiglia, F., Mylopoulos, J., Perini, A.: Tropos: An agent-oriented software development methodology. AAMAS Journal 8(3), 203–236 (2004)

3. Wooldridge, M.J., Jennings, N.R.: Software Engineering with Agents: Pitfalls and Pratfalls. IEEE Internet Computing 3(3), 20–27 (1999)
4. Wooldridge, M.: Agent-based software engineering. IEE Proc. on Software Engineering 144(1), 26–37 (1997)
5. Zhang, H.L., Leung, C.H.C., Raikundalia, G.K.: Topological analysis of AOCD-based agent networks and experimental results. Journal of Computer and System Sciences 74, 255–278 (2008)
6. Zhang, H.L., Leung, C.H.C., Raikundalia, G.K.: Classification of Intelligent Agent Network Topologies A New Topological Description Language for Agent Networks. In: Proc. of 4th IFIP Conf. on Intelligent Information Processing III, vol. 228, pp. 21–31. Springer, Heidelberg (2006)
7. Piliouras, T.C.: Network Design: Management and Technical Perspective, 2nd edn. CRC Press LLC, USA (2005)
8. Stamper, D.A.: Business Data Communications, 5th edn., pp. 183–207. Addison-Wesley Longman Publication (1999)
9. Kuchen, H., Stoltze, H.: Candela – A Topology Description Language. Journal of Computers and Artificial Intelligence 13(6), 557–676 (1994)
10. van der Ham, J.J., et al.: Using RDF to describe networks. Future Generation Computer Systems 22, 862–867 (2006)
11. Cappello, F., Desprez, F., Dayde, M., Jeannot, E., et al.: Grid5000: a nation wide experimental grid testbed. International Journal on High Performance Computing Applications (2006)
12. Dijkstra, F., Andree, B., Koyman, K., van der Ham, J., de Laat, C.: A Multi-Layer Network Model Based on ITU-T G.805. Computer Networks 52(10), 1927–1937 (2008)
13. van der Ham, J., Grosso, P., de Laat, C.: Semantics for Hybrid Networks Using the Network Description Language (2006)
14. Graham, R.L.: Peer-to-Peer: Toward a Definition. In: Proc. of International Conference on Peer-to-Peer Computing 2001, Linköpings unversitet, Sweden (2001)
15. Strogatz, S.H.: Exploring complex networks. Nature 410, 268–276 (2001)
16. Wang, X.F., Chen, G.: Complex networks: Small-world, scale-free and beyond. IEEE Circuits and Systems Magazine (2003)
17. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science 286, 509–512 (1999)

Sentiment Analysis for Effective Detection
of Cyber Bullying

Vinita Nahar[1], Sayan Unankard[1], Xue Li[1], and Chaoyi Pang[2]

[1] School of Information Technology and Electrical Engineering
The University of Queensland, Australia
[2] The Australian E-Health Research Center, CSIRO, Australia
{v.nahar,s.unankard}@uq.edu.au,xueli@itee.uq.edu.au,Chaoyi.Pang@csiro.au

Abstract. The rapid growth of social networking and gaming sites is associated with an increase of online bullying activities which, in the worst scenario, result in suicidal attempts by the victims. In this paper, we propose an effective technique to detect and rank the most influential persons (predators and victims). It simplifies the network communication problem through a proposed detection graph model. The experimental results indicate that this technique is highly accurate.

Keywords: Social Network, Cyber Bullying, Text Mining.

1 Introduction

With the proliferation of Web 2.0, cyber bullying is becoming an important issue. A number of life threatening cyber bullying experiences have been reported internationally, thus drawing attention to its negative impact [1]. Detection of online bullying and subsequent preventive measure is the main course of action to combat it.

To this end, we propose a detection method for identifying cyber bullying messages, predators and victims. Our methodology is divided into two phases. The first phase aims to accurately detect harmful messages. We present a new way of feature selection, namely common and sentiment features. The next phase aims to analyse social networks to identify predators and victims through their user interactions, and present the results in a graph model. A ranking algorithm is employed to detect the influential predators and victims. The proposed approach for anti-cyber bullying using a computed cyber bullying detection matrix and associated graphical representation of the results is unique.

Contributions. First, we propose a novel statistical detection approach, which efficiently identifies hidden bullying features to improve the performance of classifier. Second, we present a graph model to detect association between various users in the form of predators and victims. Besides identifying victims and predators, this graph model can be used to answer many important queries such as how many victims associate to the same predator. Furthermore, ranking methods are employed to identify the most influential persons on the social network. Third,

our experiments show that the proposed approach is statistically significant in terms of our test results.

The rest of this paper is organised as follows. Section 2 reviews the related literatures; Section 3 describes the proposed methodology; Section 4 explains how the experiments are designed and reports the results; Section 5 concludes this paper.

2   Related Work

Sentiment Analysis. It is a task of learning the semantic orientation of a document, sentence and phase. Sentences can also be classified as objective or subjective. Subjective sentences are ideal for sentiment classification. Extensive works have been done in this area [7]. However, the major application areas explored are product and movie reviews. Nevertheless, we employed sentiment analysis for bullying detection.

Cyber Bullying Detection. Current research to detect online sexual predators [5] correlates the theory of communication and text mining to discriminate predator and victim communication. It uses rule based approach applied on chat log dataset. Other interesting works in this area [9] and [8] mainly focused on the detection of harmful messages. [8] also performed affect analysis on inappropriate messages.

Ranking Methods. Page Rank [6] and Hyperlink-Induced Topic Search (HITS) [2] are ranking methods used extensively in many areas of network analysis and information retrieval to find appropriate hub and authority pages, where hub and authority pages are interlinked and effects each other. In both the methods, the subset of relevant web pages is constructed based on an input query, Page Rank proliferates search results through links from one to another web page to find the most authoritative page, whereas, HITS identifies authoritative as well as hub pages. Our approach uses the HITS to calculate potential predator and victim scores in the form of Eigen values and vectors.

3   Methodology

The proposed methodology is a hybrid approach. It employs sentiment analysis to classify given entry into 'bully' or 'non-bully' category and uses link analysis to find the most influential person. Each step is defined in detail as follows.

3.1   Feature Selection and Classification

The feature selection is a key step to represent data in a feature space as an input to the model. Social network's data are noisy, thus regressive preprocessing is employed. Also, the number of features grow with the number of documents. Thus, we propose following two types of feature selection methods:

A. Common Features. These are mixed features extracted from both bullying and non-bullying messages, which are generated using bag-of-word[1] approach.

B. Sentiment Features. These are generated by applying Probabilistic Latent Semantic Analysis (PLSA) [3] model on bullying posts only. Classification of text as positive, negative or neutral does not cater for the versatile nature of sentiment detection. To improve opinion identification, PLSA is used as a strong tool to deal with the hidden factors [4]. For our probabilistic approach to extract sentiment features, PLSA is applied on the messages described next to identify hidden bullying factors. Let B represent documents, which is a collection of messages $B = \{b_1,..., b_N\}$, with word $F = \{f_1,..., f_M\}$. Then the dataset can be defined as a matrix C of size N*M; $C = (n(b_i, f_j))_{ij}$, where $n(b_i, f_j)$ is the number of times f_j appears in a document b_i. Now we introduce unobserved data (latent variables) $Z = \{z_1,..., z_k\}$ within the documents. Latent variables deal with the hidden class in the documents. It is defined in the following steps: Say, $P(b)$ is a probability of a document b, $P(z|b)$ represents the probability distribution of the document over the latent class. The word generated with the probability $P(f|z)$ is conditional probability of the word given the latent class variable. The complete model in terms of word, topic and document can be defined as:

$$P(b, f) = P(b)P(f|b) \tag{1}$$

where, $P(f|b) = \sum_z P(f|z)P(z|b)$. By applying PLSA, which utilises iterative Expectation (E) and Maximization (M) steps to maximize the data likelihood, when it converges at the local optimal solution. Finally, Bayes method is applied to calculate posterior probability $P(z \to b)$ to generate feature and latent variable relationships. This information is utilized for the further classification task. These features show the stronger correlation between a word and an unobserved class; hence, a feature can be considered as a class.

Further, SVM[2] is learnt through both the features and tested for the classification of messages into one of two classes: bully or non-bully.

3.2   Victim and Predator Identification

The cyber conversation has appeared to a situation where the surge of many bullying messages toward a specific user. In a network, predators and victims are linked to each other via sent and received messages and identified by their usernames.

Scenario: We considered a subset (Figure 1) of the main network of the users. To identify the predator and victim, HITS algorithm is implemented by computing their respective scores. A predator can be identified by the highest predator score and victim by the highest victim score.

Assumption: Each user we considered is associated with at least one bullying message.

---

[1] http://en.wikipedia.org/wiki/Bag_of_words_model
[2] http://www.csie.ntu.edu.tw/ cjlin/libsvm
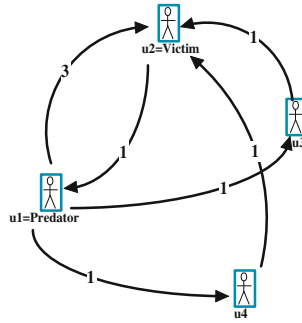
Fig. 1. Victim and Predator Identification graph

Predator: User sending out at least one bullying message.
Victim: A person who is receiving at least one bullying message.
Objective: Ranking on predators and victims to find the most influential person.
Graph model: We considered a subnet/subset (for example, four users) (Figure 1) of the main communication network of the users; which includes predators and victims. To identify the predator and associated victim, the Hyperlink-Induced Topic Search (HITS) algorithm is implemented in a victim and predator search by computing a predator and a victim score. The HITS method is based on the fact that the good hub pages point to good authority pages and good authority pages are linked by the good hub pages. The search query penetrates through web pages to identify potential hub and authority page based on the respective scores. Now, to find predator and victim, first we define following two equations:

$$p(u) \leftarrow \sum_{u \rightarrow y} v(y), v(u) \leftarrow \sum_{y \rightarrow u} p(y) \tag{2}$$

Where, $p(u)$ and $v(u)$ depicts the predator and victim scores respectively and $u \rightarrow y$ indicates the existence of the bullying massage from u to y and vice versa for $y \rightarrow u$. These two equations are an iteratively updating pair of equations for calculating predator and victim scores, which is based on our assumption defined above. In each iteration, scores are calculated through in degrees & out degrees and associated scores; this may results in large values. Thus to get the relative weights, each predator and victim scores are divided by the sum of all predator and victim scores respectively. Figure 1, delineates the identification of the predator and victim in a communication network. It is a weighted directed graph G = (U,A) where,

- Each node $u_i \in U$ is a user involved in the bullying conversation.
- Each arc $(u_i, u_j) \in$ A, is defined as a bullying message sent from u_i to u_j.
- The weight of arc (u_i, u_j), denoted as w_{ij}, is defined in the next section.

Victim and predator can be identified from the weighted directed graph G:

– The victim will be the nodes with many incoming arcs and the predator will be the nodes with many outgoing arcs. However, this paper attempts to find if a user is a potential victim or a potential predator.

3.3   Cyber Bullying Matrix

Now, to identifying predator and victim based on their respective scores, we formulate a cyber bullying matrix (w). Table 1, is a matrix w, which is a square adjacency matrix (which represents in degrees and out degrees of each node) of the subnet with entry w, which is a square adjacency matrix of the subset with entry w_{ij}, where,

$$w_{ij} = \begin{cases} n & \text{if there exist n bullying message from i to j} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

Since each user will have victim as well as predator score, scores are represented as the vectors of n*1 dimension where i^{th} coordinate of vector represent both the scores of the ith user, say p_i and v_i respectively. To calculate scores, equations $p(u)$ and $v(u)$ are simplified as the victim and predator updating matrix-vector multiplication equations. For the first iteration, p_i and v_i can initialize at 1. For each user (say, i = 1 to 4) predator and victim scores are as follows:

$$p(u_i) = w_{i1}v_1 + w_{i2}v_2 + \ldots + w_{i4}v_4, v(u_i) = w_{1i}p_1 + w_{2i}p_2 + \ldots + w_{4i}p_4 \quad (4)$$

These equations converge at a stable value to provide final predator and victim vector of each user. Finally, Eigenvectors are calculated to get both scores.

Table 1. Cyber Bullying Matrix (w)

	u_1	u_2	u_3	u_4	...
u_1	0	3	1	1	...
u_2	1	0	0	0	...
u_3	0	1	0	0	...
u_4	0	1	0	0	...
...	...	...	...	...	...

4   Experiments and Results

For our experiment, we used data available from the workshop on Content Analysis on the Web 2.0[3] and requested manually labelled data from Yin et al [9]. The dataset contains data collected from Kongregate, Slashdot and MySpace websites. Kongregate (online gaming site), provides data in the chat-log style. We have assumed that Kongregate players used aggressive words while playing.

---

[3] http://caw2.barcelonamedia.org/

Table 2. Classification performance based on common and sentiment features : MySpace and SlashDot datasets

Features	MySpace Dataset				SlashDot Dataset			
	Cases	Bully	NonBully	Accuracy %	Cases	Bully	NonBully	Accuracy%
	Sentiment features				Sentiment features			
500	740	15	725	97.97	2174	11	2163	99.49
1000	1034	28	1006	97.29	2868	18	2850	99.37
2000	1383	40	1343	97.11	3380	26	3354	99.23
4000	1657	50	1607	96.98	3808	34	3774	99.11
6000	1766	57	1709	96.77	3945	43	3902	98.91
14000	1934	65	1869	96.64	4067	48	4019	98.82
	Common feature				Common feature			
15632	1947	65	1882	96.61	4077	48	4029	98.85

Slashdot is a forum, where users broadcast messages and MySpace is a popular social network. Data from the Slashdot and MySpace were provided in the XML files, where each file represented a discussion thread containing multiple posts. Post and user information for each site were extracted and indexed through the inverted file index[4]; thus assigning an appropriate weight to each term.

4.1   Cyber Bullying Detection

LibSVM was applied for classification into bully or non-bully class using a linear kernel and 10 fold cross validation was performed. Using both the features, accuracy is reported as the evaluation criterion of three different datasets in Table 2 and 3. Table 2 defines classifier performance on Myspace and Slashdot while Table 3 shows classifier performance on Kongregate and Combined datasets.

4.2   Victim and Predator Identification

A predator and victim identification graph is developed for a given scenario. Only the messages identified as bullying are considered, as shown in Figure 1. The user information was extracted to examine victim and predator data in the format of the matrix described in Table 1. The rows depict message senders, and the columns delineate receivers. The matrix values indicate the number of messages sent and received. To analyse forum style data, we considered each user of a forum posting to be a sender and a receiver. However, we assumed that the users will not be sending message to themselves and hence assigned the message value as zero. The chat style dataset contains direct communication between two users, so there were one sender and one receiver for each message. Expert judgment is obtained by manually counting the number of sent bullying messages and ranking them. The performance of the HITS Algorithm is shown in Table 4. It shows that the HITS algorithm vs. expert judgment achieved similar predator identification accuracy, especially in regards to the top three ranked predators. Similarly, the HITS algorithm vs. expert judgment for victim identification results were also comparable. The datasets had many participants

---

[4] http://en.wikipedia.org/wiki/Inverted index

and it was not specified whether messages were sent to anyone or not. Because of this, we assumed that all users who posted on the same topic by default were a receiver too. Hence, many users received the same victim ranking, which cannot be presented in a table form. Nevertheless, the results of HITS algorithm were closely aligned to the expert judgment results.

Table 3. Classification performance based on common and sentiment features: Kongregate and Combined datasets

Features	Kongregate Dataset				Combined Datasets			
	Cases	Bully	NonBully	Accuracy %	Cases	Bully	NonBully	Accuracy%
	Sentiment features				Sentiment features			
500	356	0	356	100.00	3240	26	3214	99.20
1000	575	2	573	99.65	4477	48	4429	98.69
2000	979	9	970	98.98	5742	75	5667	98.68
4000	1772	15	1757	99.10	7237	99	7138	98.63
6000	2276	23	2253	99.08	7987	123	7864	98.49
14000	3995	42	3953	99.22	9996	155	9841	98.54
	Common feature				Common feature			
15632	4292	42	4250	99.25	10316	155	10161	98.55

Table 4. Performance of Ranking Method : Predators

Expert Judgement		Hits Algorithm	
Rank	User id	Rank	User id
1	Android457	1	Android457
2	AmberPeace	2	AmberPeace
2	chance10149	2	chance10149
3	MS_143549239	3	YOUAREAHOMO
3	MS_226698859	3	mock03
4	mock03	4	im69
4	MS_3449431	4	Komedia
4	YOUAREAHOMO	4	LordShadow

4.3   Discussion

The performance of the classifier is evaluated on its accuracy based on both the features, as tested on the three different datasets. Accuracy is defined as the degree of measure by which classifier accurately identifies the true value. As shown in the graph (Figure 2), the results indicate a very high accuracy of the classifier with sentiment features, especially when the features are in the range of 500 to 4000, while it remains almost constant between 6000 to 14000. The classifier performs best with 500 features. We identified the top ranked 764 features with the best accuracy results as selected by the PLSA. We selected the top 500 features and calculated the accuracy for each case. We found that the F-1 measure is not significant for our evaluation, because of the large number of negative cases. We chose feature selection methods that optimised the accuracy of the classifier and achieved very high accuracy outcomes. We proposed a weighted directed graph model to critically analyse and answer user queries regarding victims and predators. Based on the weighted arcs between two users, the model iteratively assigns the victim and the predator score to each user and correctly identifies the most influential predator and victim.
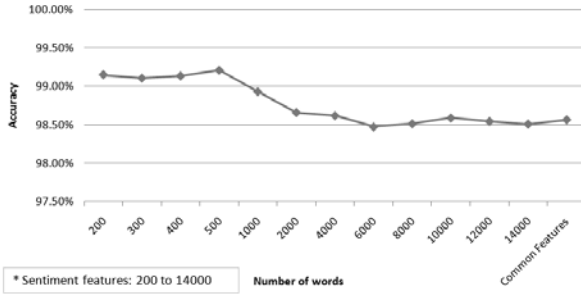
Fig. 2. Overall Performance based on Accuracy

5   Conclusions

We presented an effective sentiment analysis technique in this paper to detect cyber bullying messages by using PLSA for feature selection. The experimental results show that feature selection improves the accuracy of classifier, thus reducing resource usage significantly. In addition, we employs the HITS algorithm to calculate scores and rank the most influential persons (predators or victims). The proposed graph based model can be used to answer various queries about the user in terms of bullying. In future research, we plan to continue the in-depth analysis of indirect bullying and its emerging patterns, to help in its identification and prevention of cyber bullying.

References

1. ncpc.org, http://www.ncpc.org/cyberbullying (accessed September 15, 2011)
2. Easley, D., Kleinberg, J.: Link analysis using hubs and authorities. In: Networks, Crowds, and Markets: Reasoning about a Highly Connected World, pp. 399–405
3. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. Machine Learning 42, 177–196 (2001)
4. Liu, Y., Huang, X., Aijun, A., Yu, X.: Arsa: a sentiment-aware model for predicting sales performance using blogs. In: Research and Development in Information Retrieval, SIGIR, pp. 607–614 (July 2007)
5. McGhee, I., Bayzick, J., Kontostathis, A., Edwards, L., McBride, A., Jakubowski, E.: Learning to Identify Internet Sexual Predation. International Journal on Electronic Commerce 15(3), 103–122 (2011)
6. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab (November 1999)
7. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proc of the ACL, pp. 271–278 (2004)
8. Ptaszynski, M., Dybala, P., Matsuba, T., Masui, F., Rzepka, R., Araki, K.: Machine Learning and Affect Analysis Against Cyber-Bullying. In: Proceedings of the Thirty Sixth Annual Convention of the Society for the Study of Artificial Intelligence and Simulation of Behaviour (AISB 2010), March 29- April 1, pp. 7–16 (2010)
9. Yin, D., Xue, Z., Hong, L., Davisoni, B.D., Kontostathis, A., Edwards, L.: Detection of harassment on web 2.0. In: Content Analysis in the WEB 2.0 (CAW2.0) Workshop at WWW 2009 (April 2009)

Bizard: An Online Multi-dimensional Data Analysis Visualization Tool

Zhuoluo Yang[1,2], Jinguo You[1,2], Jian Wang[1], and Jianhua Hu[1]

[1] School of Information Engineering and Automation,
Kunming University of Science and Technology,
Kunming, Yunnan, China
[2] Yunnan Computer Technology Application Key Lab,
Kunming University of Science and Technology,
Kunming, Yunnan, China

Abstract. In order to address the visualized analysis on the multi-dimensional data, this demo paper designs and implements a visualization analysis tool, named as *Bizard*. *Bizard* encapsulates the underlying Hadoop Client or XML/A data access interface, which makes it convenient to use conventional analytical tools. By Rich Internet Application (RIA) technology, *Bizard* designs multiple visualization dashboard widgets and supports interactive multi-dimensional analysis.

Keywords: Hadoop, Multi-dimensional data, OLAP, Visualization.

1 Introduction

As Internet grows to the enormous size, massive amounts of data are needed for the ad-hoc online analysis and data visualization. Data visualization makes large-scale data comprehensible and available for decision making. In conclusion, it is not only essential to visualize enormous data, but it is also necessary to develop effective visualization technology and tools.

Data analysis systems such as Micro Strategy, Business Objects, ArcPlan, Oracle BIEE, Cognos, PowerPlay and Dundas are suitable for relational analysis and rather strict in reporting and analyzing clients. They support neither extremely large data sets (TB Scale) nor XML/A standards [1].

A web based online analysis processing system named as *Bizard* has been proposed. It supports the Hadoop [2] or the XML/A data source. *Bizard* is built on top of the Java EE platform. The application is distributed and supports various clients. A Rich Internet Application (RIA) online analysis has also been proposed to allow clients to interact with the Java EE application.

2 System Architecture

The system can be divided into three layers consists of *Storage* at the base, followed by *Service*, and *Application* consecutively. As illustrated in Fig. 1, the
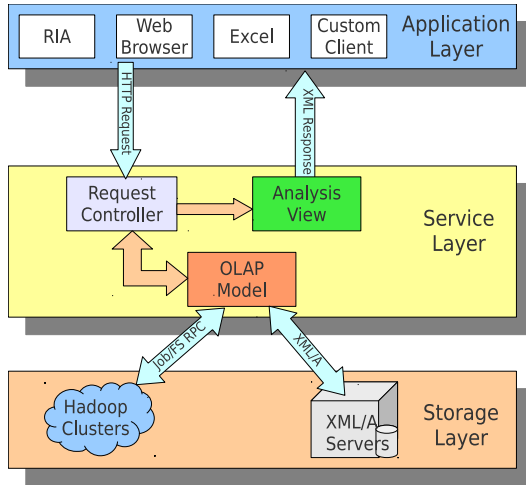
Fig. 1. System Architecture

Storage Layer accesses the data from either XML/A servers or Hadoop Distributed File Systems (HDFS) [3]. In the cluster scenarios, the *Storage Layer* uses MapReduce [4] paradigm to generate data cubes [5]. MapReduce algorithms are good at computation of locality and parallelism. The storage layer wraps both the Hadoop Filesystem Client and the Hadoop Job Client. The *Service Layer* consists of a group of Tomcats with an NginX, which are used for request distribution and load balance.

2.1   Storage Layer

The *Storage Layer* can be either XML/A servers, such as Mondrian or Microsoft SQL Server for Analysis (SSAS), or Hadoop clusters, such as HDFS or higher level Pig [6] and Hive [7]. For simplicity, the *Storage Layer* only supports these two kinds of data sources. The OLAP Model of the upper layer accesses the XML/A server by XML/A protocols and accesses the Hadoop cluster by Remote Procedure Calls (RPC).

However, there is no coupling between the XML/A server and the Hadoop cluster. The OLAP Model in the *Service Layer* accesses the Hadoop cluster by the Filesystem/Job client and accesses the XML/A server by the XML/A client. These two kinds of clients are encapsulated by the OLAP Model of the upper layer.

2.2   Service Layer

The *Service Layer* consists of a group of Tomcat containers and an NginX server. Each container holds a Java EE application based on the MVC architecture [8]. The NginX server is used to distribute requests to the distributed Java EE

applications, so that the load of containers can be easily balanced. The NginX also processes all the static resource requests, such as script files and pictures, reduces the static resource processing time.

In the Java EE applications, the controllers only process computation needed HTTP requests which access the specific data of the *Storage Layer*. While processing an HTTP query from the clients, the controller processes the request and generates a command of the model. The model retrieves the data from the *Storage Layer*, generates an OLAP model instance in the memory and sends the reference of the OLAP model object to the controller. The controller serializes the object into an XML of the specific format and sends it to the analysis view. The view constructs the XML file stream and sends it to the clients by HTTP response. So the HTTP request and the XML response form an easy to use API for custom clients.

2.3   Application Layer

The *Application Layer* can be any HTTP clients, for example RIAs and Web Browsers etc. Usually, the client is a web browser which supports W3C standards well. We have built an RIA web client by jQuery and Flex. The jQuery script generates the HTTP request, sends it to the controller of the *Service Layer* and processes the XML response of the view. After processing the XML response, the jQuery script renders the XML to an XHTML through a remote fetched XSL and generates the colorful web pages, as seen in Fig. 2.

The upper left and lower right widgets in Fig. 2 are built by Flex and the upper right and lower left are rendered by jQuery. The four widgets can interact together as shown in this figure. The external interface of the Flex is used to
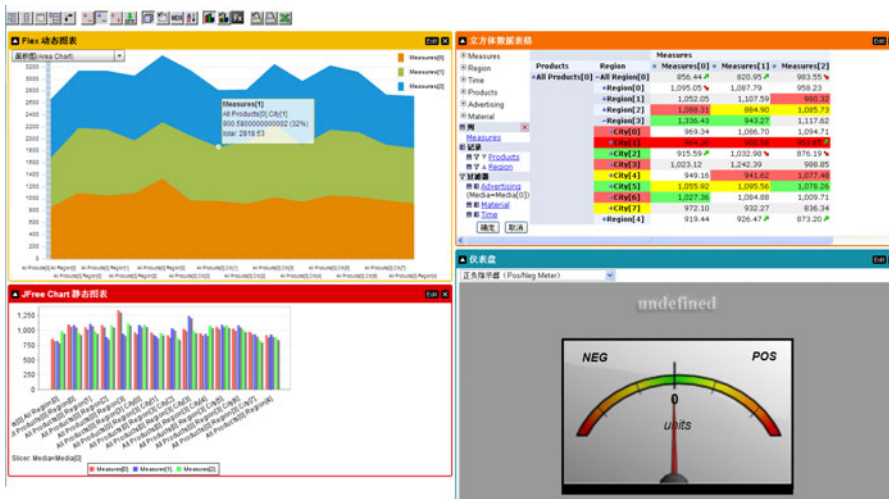
Fig. 2. The RIA Screenshot

interact with the jQuery script, so the communication between the RIA and the Java EE Application is done by the jQuery only once. We make use of client CPUs to render the charts and tables, hold only a small copy of aggregation data in the client memories and ease the load of the servers.

3   Conclusions and Future Works

We have proposed an online data processing and analysis system named as *Bizard*. The main contributions of *Bizard* are:

(1) The construction of data model which encapsulates XML/A clients and Hadoop Job/Filesystem clients.

(2) A distributed online analysis Java EE application is built on the open sourced Tomcat and NginX.

(3) A powerful and efficient RIA client for the online analysis server was created and also has been proposed as an easy to use API to the custom clients. XML response data will be compressed in the future to fully use client CPU and allow for manipulation of client data in a much larger scale. More powerful and easy to use clients will also be created for the system.

References

1. Tang, Z., Maclennan, J., Kim, P.P.: Building Data Mining Solutions with OLE DB for DM and XML for Analysis. ACM SIGMOD Record 34(2), 80–85 (2005)
2. Apache Hadoop, `http://hadoop.apache.org`
3. Borthakur, D.: The Hadoop Distributed File System: Architecture and Design. The Apache Software Foundation (2007)
4. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM 51(1), 107–113 (2008)
5. Nandi, A., Yu, C., Bohannon, P., Ramakrishnan, R.: Distributed Cube Materialization on Holistic Measures. In: 2011 IEEE 27th International Conference on Data Engineering (ICDE), pp. 183–194. IEEE Press (2011)
6. Olston, C., Reed, B., Srivastava, U., Kumar, R., Tomkins, A.: Pig Latin: A Not-so-foreign Language for Data Processing. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1099–1110 (2008)
7. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Zhang, N., Antony, S., Liu, H., Murthy, R.: Hive - A Petabyte Scale Data Warehouse Using Hadoop. In: 2010 IEEE 26th International Conference on Data Engineering (ICDE), pp. 996–1005. IEEE Press (2010)
8. Leff, A., Rayfield, J.T.: Web-application Development Using the Model / View / Controller Design Pattern. In: Enterprise Distributed Object Computing Conference, pp. 118–127. IEEE Press (2001)

Jingwei+: A Distributed Large-Scale RDF Data Server

Xin Wang[1], Longxiang Jiang[2,*], Hong Shi[1], Zhiyong Feng[1], and Pufeng Du[1]

[1] School of Computer Science and Technology, Tianjin University, Tianjin, China
{wangx,serena,zyfeng,pdu}@tju.edu.cn
[2] School of Computer Software, Tianjin University, Tianjin, China
lxjiang2012@gmail.com

Abstract. With the development of the Linked Data project, the amount of RDF data published on the Web is steadily growing. To deal with the management issues of large-scale RDF data efficiently, we have developed a Bigtable-based distributed RDF repository, named Jingwei+, with a high level of scalability, to support the fast execution of triple pattern queries. The data service engine of Jingwei+ provides RESTful API to the outside, and its Web user interface have implemented the triple pattern query and the Linked Data navigational browsing. This paper demonstrates the query and navigation features of Jingwei+. The Jingwei+ RDF data server has laid the foundation of further development of the Semantic Web search engine.

Keywords: large-scale RDF data, distributed data server, storage scheme.

1 Introduction

RDF (Resource Description Framework) [1] has become the standard data model for representing and exchanging the machine-understandable information on the Semantic Web. Each Web resource is identified by an HTTP URI. RDF dataset is a finite set of triples (S, P, O), and each triple is expressed as three terms in which S occurs as subject, P as predicate and O as object. The subject, predicate and object can be represented as an HTTP URI, and the object can also be a literal or blank term. Fig. 1(a) shows an example of RDF triples.

With the development of the Linked Data project, more and more RDF data has been released on the Web. By September of 2011, the amount of triples on the Web has reached 31 billion [2]. Such large-scale RDF data and the inherent flexibility of RDF have posed new challenges to traditional data management systems. In order to manage these large-scale RDF data efficiently, we have developed a distributed RDF data server called Jingwei+. Firstly, Jingwei+ achieves the horizontal scalability and the high efficiency of RDF querying and processing by using the multi-dimensional nested key-value store. Secondly, the storage performance is promoted via namespaces transformation and compression. Thirdly, the user requests are fulfilled by the fast execution of triple pattern queries and the navigational browsing interface for the Linked Data. Fig. 2. shows the system architecture of Jingwei+.

---

* Corresponding author.

2   Features

This section presents the core features of Jingwei+ including RDF storage scheme based on Bigtable, namespace transformation and compression, and triple pattern queries.

2.1   RDF Storage Scheme Based on Bigtable

Bigtable is a distributed storage system for managing structured data that is designed to scale to a very large size [3]. Our storage scheme is based on an extended Bigtable storage model which is a sparse, distributed and persistent storage scheme using the multi-dimensional sorted map. There are 5 levels of the mapping structure in our extended Bigtable storage scheme including K, CF, SC, C and V (underlined letters in Fig. 1(b)). Actually, we use K for storing S, SC for P and C for O as depicted in Fig. 1(b). To improve the query performance, the RDF triples are indexed in three different orders. For instance, SPO means we index RDF triples in the order of subject, predicate and object. There are three indexes: SPO, POS and OSP. We can get higher performance through trading space for time by using different indexes.

(a)   (b)
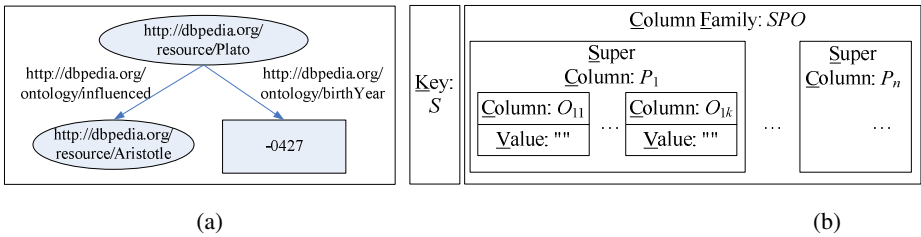
Fig. 1. (a) Example of RDF triples; (b) Extended Bigtable storage model for RDF
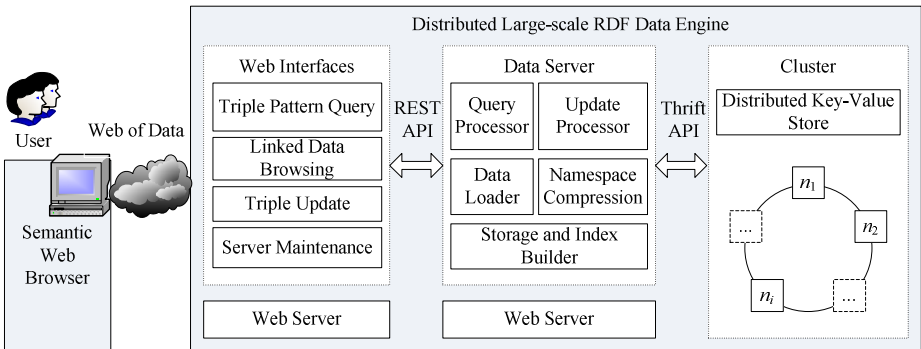
Fig. 2. System architecture of Jingwei+

2.2   Namespace Transformation and Compression

According to the RDF standard [1] and the principles of Linked Data [5], an RDF data set contains a large number of HTTP URI strings, and many of them share the same URI prefix. In order to reduce the storage space and improve the query performance, we transform high-frequency prefixes into simple abbreviations, thus compressing the URI strings.

2.3   Triple Pattern Query

A triple pattern query, which is the most fundamental component of an RDF query, is similar to an RDF triple except that it places the character "?" in front of a term to indicate a variable. In fact, triple pattern queries are essentially a subset of the SPARQL query language [4]. Therefore, triple pattern queries require an efficient processing mechanism. On the basis of our storage and indexing scheme, Jingwei+ selects the corresponding index for a query according to the variable occurrences.

3   Demonstration

This section presents the demonstration environment and scenarios of Jingwei+.

3.1   Demonstration Environment

Jingwei+ is written in Java, and the version of JDK is 1.6. There is an 8-nodes cluster in the underlying storage system. Each node has a 2.4 GHz CPU and 2 GB memory. The operating system is Ubuntu 10.04. We use Apache Tomcat 7.0 as the Web server. Currently, Jingwei+ has loaded two datasets: DBpedia [6] including ontology mapping based instances (13.8 million triples), short abstracts of DBpedia, images links and extended links to Geonames [7], and Geonames 2.2.1 RDF dump (145.9 million triples).

3.2   Demonstration Scenarios

In this section, we present the demonstration scenarios by the following use cases.

Use case 1: Triple pattern query. Input 7 different types of triple pattern queries in the Jingwei+ Web interface. Table 1 shows these query results and execution times. Fig. 3(a) shows the result of query t5.

Use case 2: Linked Data navigational browsing. On the basis of query t5, we can get more detail information if we click on the object Aristotle as shown in Fig. 3(a). Any resource can be displayed as long as a corresponding HTTP URI is available.

Use case 3: Query across datasets. Select a dataset before executing a triple pattern query. We can issue a query involving DBpedia URIs when the Geonames dataset is selected. Fig. 3(b) shows the results of querying Beijing (using the DBpedia URI) in Geonames. The default option is "All Datasets".

Table 1. Demonstration results for triple pattern query

Use case number	Triple pattern query	Number of results	Execution time / s
t1	(r:Plato, o:influenced, r:Aristotle)	1	0.005
t2	(r:Plato, o:mainInterest, ?X)	10	0.007
t3	(r:Plato, ?X, r:Aristotle)	1	0.008
t4	(?X, o:influencedBy, r:Plato)	79	0.011
t5	(r:Plato, ?X, ?Y)	52	0.006
t6	(?X, ?Y, r:Plato)	88	0.017
t7	(?X, o:mainInterest, ?Y)	2240	0.149

(a)   (b)

(c)

Fig. 3. (a) Triple pattern query and navigational browsing; (b) Query across datasets; (c) RDF triple update

Use case 4: RDF triple update. Input an original triple (r:Aristotle, o:mainInterest, r:Physics) and a new triple (r:Aristotle, o:mainInterest, r:Football). Then execute the update operation. Fig. 3(c) shows the screenshot of the update.

4   Conclusion

This paper describes the architecture of a novel distributed large-scale RDF data server called Jingwei+, and demonstrates its key features. In the future, many aspects of the system can be improved, such as the design of an inverted list for RDF keyword search and the analytical queries on large-scale RDF data using MapReduce. Also, we will conduct experiments to evaluate performances on Jingwei+ with other Linked Data datasets besides DBpedia and Geonames, and compare Jingwei+ with other related systems.

References

1. Klyne, G., Carroll, J.J., McBride, B.: Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, World Wide Web Consortium (2004)
2. Linked Data, `http://linkeddata.org/`
3. Chang, F., Dean, J., Ghemawat, S., Hsieh, C.W., Wallach, A.D., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A Distributed Storage System for Structured Data. In: 7th USENIX Symposium on Operating Systems Design and Implementation, pp. 205–218. USENIX Association, Berkeley (2006)
4. Prud'Hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommendation, World Wide Web Consortium (2008)
5. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems 5(3), 1–22 (2009)
6. DBpedia, `http://dbpedia.org/`
7. Geonames, `http://www.geonames.org/`

Baby Careware: A Online Secured Health Consultant

Yanjun Cui[1,2], Yanping Zhou[1,2],
Huan Mei[1,2], and Zheng Zhao[3]

[1] HeBei Applied Mathematics Institute,
46 South Youyi Street, Shijiazhuang, China
{CuiYanjun,ZhouYanping,MeiHuan}@heb-math.org
[2] Hebei Authentication Technology Engineering Research Center,
[3] Tianjin University,
92 Weijin Road, Nankai District, Tianjin, China
ZhengZh@tju.edu.cn

Abstract. We provide a design and implementation of a mobile health system which allows the parent to ensure their babies health and obtain advice from the doctor remotely. This system is based on Android platform and uses PKI technology to protect the user's privacy. The parents can upload their baby's physical growth value to the server, obtaining the curves to contrast with the standard health indicators. The parents can specify who are allowed to browse their baby's information and can establish a link with other babies with the same interests. All physical growth values are encrypted with the devise we developed, which is supported by Smart phone's TF interface and cryptographic algorithms.

Keywords: Android, Cryptograph, TF Security Card, Privacy protection.

1 Introduction

For the one-child policy in China, children have become the centre of families in urban areas. The parents' focus has shifted from food to every aspect of the baby. Parents want the knowledge of their baby's development status and obtain professional advice directly. As the development of 3G network in China, smart phone has become a common tool for most people. According to Google's reports, penetration of smart phone in China's urban area has reached to 35% in 2011. With the spread of the smart phone and reduction of the 3G network cost, it is possible to create a network application to help the parents to raise their baby scientifically.

Our project will satisfy the needs of the parents: recording the baby's growth values, checking the baby's growth status and getting advices from doctors.

To achieve this vision, there are 5 basic principles we need to follow.

☐ Security - The babies' physical growth value need to be protected from leaking;

☐ Convenience - The parents can upload babies' growth value and get advices conveniently;

☐ Visualization- The analysis result of the babies' growth value can be checked visually and directly;

☐ Profession - The analysis should be base on professional evaluate model;

☐ Interaction - The interaction of parent - parent and parent - doctor should be easily.

In this paper, we describe our efforts to provide self protecting the Child Development File (CFD) [1] using recent developed TF interface security devices combined with cipher-based attribute based encryption (CP-ABE). Our prototype runs as an android app[6] and all the data uploaded to server can only be accessed by the one who has the attributes to retrieve the keys used to encrypt the data[5]. The primary contributions of this project are:

☐ We designed a TF interface security card that can be used in smart phones. This device can replace the common TF card we just used. Beside the storage function, it provides extra functions: Digital Signature, Encryption, Digital digest etc.

☐ We hired several popular evaluation models such as BSID, GDDS, DDST to allow the parent to check the babies' growth statics in multiple aspects.

1.1 TF Security Card

Limited by the power of CPU, the smart phones we used are not suitable for cryptographic calculation. Cryptographic calculating is both time and space consuming. In order to satisfy the need of high performing calculations, we designed a device that can separate the cryptographic calculating from the smart phone. We designed and created a compound device that supports the storage function and cryptographic functions, since all smart phones are created to support TF interface nowadays. This device is called the TF Security card.

1.2 Key Exchange

With the TF security card, the user can enroll a digital certificate from CA and use it as a network Id. Every time the user request data from the server, the server create a digital envelop to exchange data. The principle of digital envelope can be seen in figure 1. Thus, it guarantees that the symmetric key uses only once in each session.
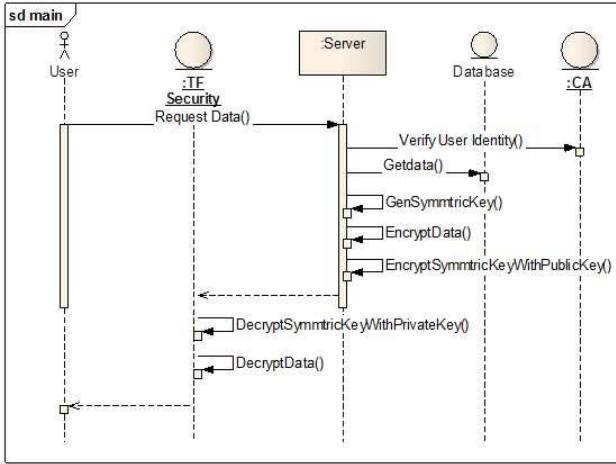
Fig. 1. Digital Envolop

2   System Overview

This project we called Baby Careware consists of two main parts: the server and the client. The server part of this project can be placed in the Cloud and be accessible through the internet. It stores data and give the evaluation result of the babies. On the other hand, the client part of this project is a smart phone running android system. The parents can upload the babies' growth values, receive evaluation result and get personalized guidance through it.

As a Cloud application, the key factor of the application is security. Our project uses an execution server to prevent the user from accessing the babies' data directly. The overall structure of the project can be seen in figure 2 below.
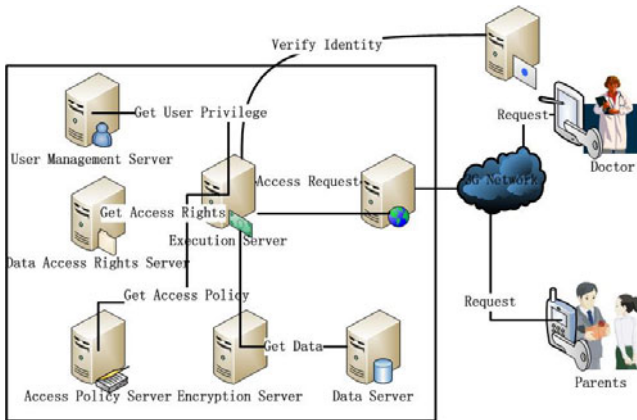
Fig. 2. System Overview

When a user need to access the babies' data, the request redirect to the execution server. The execution server then collects information about the user and the data being accessed. First, the user's identification should be verified by the user's certificate with the third part CA. Once the user's identification is confirmed, the execution server retrieves the user's attributes from the user's management server and access privilege from the data access rights server, and then once again verifies if the access fulfils the access policy. Finally, the execution server notify the encryption server creating a digital envelop to send data to the user. All the data transferred in XML format as discussed in [2].

3   Conclusion

Electronic health records management has been discuss by [1][2][3] and [4], but none of them support strong encryption and identity verification on the client's side. The Baby Careware provides the parents with intuitive and direct view about their babies' health and intelligence development level, and guides them bringing up the infants scientifically. Our next step is to build a social network to help the parents exchange experiences on bringing up children. All data flow in the system is protected by cryptographic algorithms, that make the whole system safer than ever.

References

1. Akinyele, J.A., et al.: Securing Electronic Medical Records Using Attribute-Based Encryption On Mobile Devices. ACM (2011) 978-1-4503-1000–/11/10
2. Mohammed, S., Fiadhi, J., Mohammed, O.: Securely Streaming SVG WEB-Based Electronic Healthcare Records involving Android Mobile Clients. Journal of Emerging Technologies in Web Intelligence 11(2) (November 2009)
3. Ahn, G.-J., Mohan, B.: Role-Based authorization in decentralized health care environment. In: 18th ACM on Applied Computing (2003)
4. ASTM International. ASTM E2369-05e1 Standard Specification for Continuity of Care Record (CCR) (2009)
5. Becker, M.Y., Cassandra, P.S.: Flexible trust management, applied to electronic health records. In: 17th IEEE CSFW (2004)
6. Android developers guid, `http://www.developer.android.com/index.html`

Mobile Applications towards Prevention and Management of Chronic Diseases[*]

Hang Ding, Marlien Varnfield, and Mohan Karunanithi

The Australian e-Health Research Centre, Level 5, UQ Health Sciences Building 901/16,
RBWH, Herston, QLD, 4029, Australia
hang.ding@csiro.au

Abstract. Chronic disease is the leading cause of death and disability, and poses a major burden to heathcare systems in Australia and other western countries. To alleviate this burden, we developed three mobile phone based solutions to address primary and secondary prevention of the leading chronic diseases such as cardiovascular diseases and chronic obstructive pulmonary diseases. This paper focuses on how the mobile solutions could address the clinical problems, and briefly discusses some of the preliminary findings.

Keywords: mobile health, cardiovascular disease (CVD), chronic obstructive pulmonary disease (COPD), physical activity.

1   Introduction

Due to the ageing population, modern lifestyle, urbanization and industrialization there is an increased prevalence of chronic disease globally. Chronic disease has been responsible for 63% (36 million) of global deaths in 2008 [1]. In Australia, chronic disease is the leading cause of death and disability [2]; and affects over 7 million Australians (30% of population). They were responsible for 70% ($A34 billion) of overall disease expenditure in 2001[3]; and were projected to reach 80% in 2020 [4]. Although Australia's health care system ranks as one of the best among OECD countries, it currently is under a great pressure of increasing cost, crowed hospitals, low staffing levels, overloaded emergency departments, and long waiting lists on the elective surgery. In response to these emerging problems, the Australian government took the initiative in 2006 in directing effective prevention and management of chronic diseases as a key policy objective of the Australian and all state and territory health systems [7], hence significant moves have been initiated to extend care in the community and home.

To enable this initiative in the community care and maintain a similar level of care and connectivity with clinicians, the Australian e-Health Research Centre, in collaboration with Queensland Health, has developed mobile homecare models and applications to assist prevent and manage chronic diseases. This paper demonstrates three

---

[*] Corresponding author.

research applications of mobile healthcare being trialed in Australia targeting: 1) secondary prevention of cardiovascular diseases (CVDs), 2) management of Chronic Obstructive Pulmonary Disease (COPD), and 3) health promotion programs for regional Queensland. As the trials have not been completed, the paper only focuses on how the mobile solutions address the clinical problems, and briefly discussed some preliminary analysis results.

2   Homecare Model for the Secondary Prevention of CVDs

Cardiovascular Disease (CVD) is the leading chronic disease burden to the Australian healthcare system and worldwide [6]. The secondary prevention of CVDs through cardiac rehabilitation (CR) is found to be beneficial in reducing the risks of the mortality, morbidity, and disability of CVDs.   However, the uptake of traditional CR programs through centre- or hospital- based setting in Australia is very poor (16%-19% of eligible patients). To improve the uptake of CR programs, an ICT (mobile phone and internet) –based home care model, known as the Care Assessment Platform (CAP), was developed to deliver CR services at home. The setup of the CAP is shown in Fig.1.
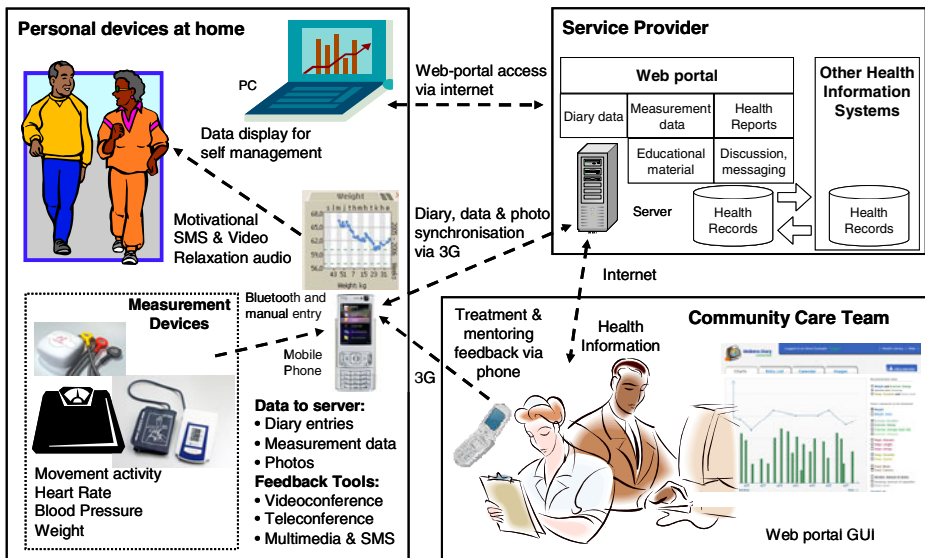
Fig. 1. Care Assessment Platform, a mobile and internet based care model for the secondary prevention of cardiovascular diseases

Through the CAP, patients input their observed (smoking, alcohol, sleeping quality, diet, physical activities, and tiredness) and measured medical/health (body weight, blood pressure, and blood glucose levels) data by using the mobile phone on a daily basis. An in-built motion sensor and application within the mobile phone

automatically measure patients' daily physical activity. These data on the mobile phone are synchronized daily to a centralized server. These are then delivered to the Clinicians or CR coordinators to review the patients' data via an internet, and provide weekly consultations and interventions to patients undergoing CR at home. Education and motivational materials are delivered through multimedia (videos and audio) preloaded on the phone. Corresponding daily SMS messages are also sent automatically to the patients.

Currently, the CAP is being validated in a clinical randomized controlled trial (RCT). In the trial, patients (n=120) from the Metro North Health Services District, Queensland, were randomised to the CAP (n=60) and control, hospital based (CT) based (n=60), CR groups. The trial will provide evidence on the clinical and economic outcomes. Some preliminary analysis results demonstrated that the CAP significantly ($p<0.05$, Fisher exact test) reduced the dropouts of patients attending CR programs. These were mainly due to life demands (work and family commitments) (21.7% (CT) vs. 5.0% (CAP)) and facilitating factors (difficulties in transportation and/or parking) (21.7% (CT) to 3.3% (CAP)). The operating cost per person was also reduced from A\$2245 (CT) to A\$1720 (CAP). This reduction will translate to further cost savings with increasing number of patients, as infrastructure costs with the CAP home care model become considerably reduced, compared with the hospital based program. In addition, the trial is expected to create new clinical knowledge on physiological signal patterns, exercise, diet, lifestyle, and behavior of cardiovascular patients in the homecare setting.

3   Homecare Model for Chronic Obstructive Pulmonary Disease

Based on the CAP, a mobile-web based home model (M-COPD) was developed to assist clinicians to monitor and manage patients with COPD at home. COPD is a major cause of morbidity and mortality, and will become the third leading cause of death worldwide in 2020 [7]. In Australia, COPD was estimated to affect 2 million people in 2008 and is expected to affect 4.5 million in 2050 [8].

M-COPD can assist clinicians monitor the major symptoms of COPD and vital signs of patients at home, and provide early diagnosis and treatment of COPD exacerbations (increased severity of COPD). In the M-COPD program, patients use the mobile phone to update self assessment and observation data relating to their sputum color, wheezing, cough, and vital signs, etc. The corresponding graphical plots of these data are made available to clinicians via an internet portal to assess the patients' disease progress, and clinicians provide consultations and interventions, accordingly, through mobile media and communication tools. The generic mobile web application can virtually be accessed by any web browser of smart phones, tablet PCs, and other mobile devices. In addition to the disease management, many new components, including patient's administration, data visualization, automatic data analysis, data import and export, and clinical document tracking, are integrated in the model for the management and data analysis of a pilot clinical research study at the Royal Perth Hospital in Western Australia.

4 Application for Health Promotion in Queensland

Physical inactivity has been identified as a leading risk factor for global mortality and many diseases including CVDs, diabetes, cancer, and mental diseases [9]. Prevalence of physical inactivity has been alerted in the Capricorn region in Queensland. Researchers have reported that, due to the high density of mining industries in the region, male population are generally young (median age of 34 years), but have a high body weight, low physical activity, and poor nutrition. To promote exercise and healthy eating in men in the Capricorn region, a primary health promotion program (ManUp) was developed. ManUp was designed to motivate and educate men to have active life style and healthy eating behaviours through integrated social networks, online self exercise management environment, exercise challenging schemes, and automatic exercise monitoring and reports. Currently ManUp is undergoing a RCT, where the mobile phone and control (education through traditional printed promotion materials) groups will be compared and evaluated in terms of adherence to exercise, healthy eating habits, and reduction in body weight.

5 Summary

Chronic diseases have led to a prominent threat to the quality of life in Australia and other developed nations. This has placed a severe burden on the Australian health care system. Mobile phone based healthcare programs are becoming attractive towards the primary and secondary prevention and home care models for management of chronic diseases. Mobile health solutions are fairly new and are associated with technical challenges of usability, interoperability, and sustainability. Hence, their reliability and adaptability to provide the quality of service and user/clinician acceptance, respectively, are paramount and important to be tested for evidence in a controlled clinical environment within medical guidelines and care protocols. The Australian e-Health Research Centre is leading in the development and clinical validation of innovative mobile-based homecare and prevention programs towards providing evidence based prevention and management of chronic diseases.

References

1. Global status report on noncommunicable diseases 2010, WHO (2010)
2. National Health Survey, Australia (2004/2005)
3. Health and Welfare series No 21, AIHWC Cat No HWE-28. AIHW, Canberra (2005)
4. http://www.health.gov.au/internet/main/publishing.nsf/content/chronic
5. National Chronic Disease Strategy, National Health Priority Action Council (NHPAC), Canberra (2006)
6. Priority medicines for Europe and the world. WHO (2004)
7. Lopez, A.D., Shibuya, K., Rao, C., et al.: Chronic obstructive pulmonary disease: current burden and future projections. Eur. Respir. J. 27, 397–412 (2006)
8. Economic impact of COPD and cost effective solutions. The Australian Lung Foundation (2008)
9. Global Recommendations on Physical Activity for Health, WHO (2010)

A Healthcare Information System with Augmented Access Controls

Nagajyothi Gunti[1], Weiqing Sun[2], Mingzhe Xu[2], Zidong Liu[1],
Mohammed Niamat[1], and Mansoor Alam[1]

[1] Department of EECS, The University of Toledo, Ohio, USA
{nagajyothi.gunti,zidong.liu,mohammed.niamat,
mansoor.alam2}@utoledo.edu
[2] Department of ET, The University of Toledo, Ohio, USA
{weiqing.sun,mingzhe.xu}@utoledo.edu

Abstract. In the healthcare industry, the old paper-based record is becoming a thing of the past and more and more patient information is being transferred into the digital format, that is, Electronic Medical Record (EMR). It integrates heterogeneous information within the Healthcare Information Systems (HIS) stressing the need for augmented security, availability and access controls. We demonstrate our prototype system that incorporates the isolation and delegation components based on the real world HIS software OpenEMR. This system is targeted at enhancing the usability of contemporary HISs without degrading the system security.

1   Introduction

In today's world, technology is constantly changing which reshaped the healthcare and revolutionized the medical profession. The healthcare industry is currently undergoing a gradual migration from the old paper-based patient record system to the Electronic Medical Record (EMR) system. However, protecting the security of these patient records requires a solid infrastructure and a proven security mechanism. Access control is at the heart of the Healthcare Information System (HIS) as it is the key technique to protect and make efficient use of the vast amount of electronically stored medical information. Although traditional Role-based Access Control Model (RBAC) [1, 2] employed by most HISs provides security against unauthorized accesses, it also causes usability issues. For instance, intern doctors during training cannot access the HIS without the supervision of the doctors. And such a strict access control has led to the tragic event [3].

We design our system to accommodate the special access requirements for the medical professionals in HIS which will not be granted as dictated by the traditional RBAC. The rest of the paper is organized as follows. Section 2 provides an overview of our system. Section 3 describes the implementation and evaluation of the system. Finally, we conclude in Section 4.

2 System Overview

We designed our system as shown in Figure 1 by incorporating the access control mechanism based on two models: I-RBAC model [4] and Role-Delegation model [5]. By using the I-RBAC model, the operation on the object by the role is executed inside an isolation environment if the role or operation is predefined to be isolated. This enables the system administrators to specify the roles that need to be first isolated and then checked for security and consistency violations. Such special roles can include new employees, intern doctors and others. In the Role-Delegation model, a junior role can be temporarily granted the senior role's permissions by means of delegation. However, the junior role is not allowed to take the permissions which are only granted to the senior role. This can be especially useful to accommodate emergency or unusual situations in HIS. To make it more secure, after the delegation request is granted, our system will redirect the operations from the delegatee to access the isolated patient records and use the security check to ensure that the operations would not lead to security or consistency violations.
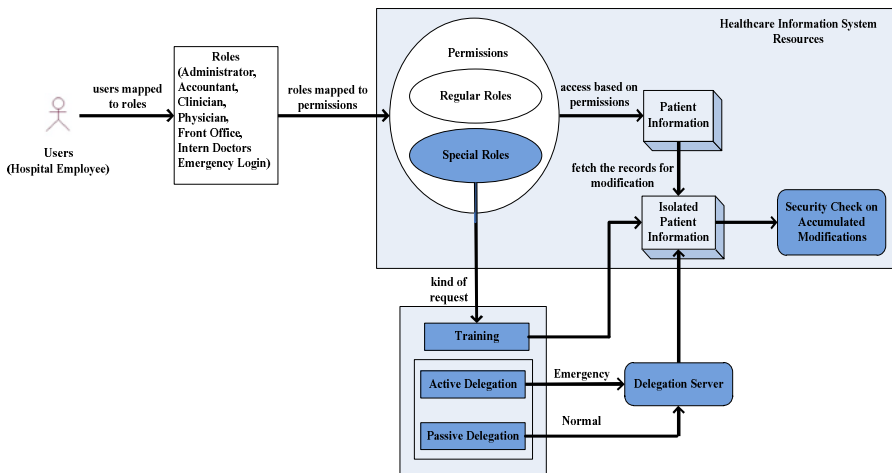
Fig. 1. Augmented Access Controls in Our System

3 System Implementation and Evaluation

Our system was implemented based on OpenEMR. It is a full featured electronic medical record and practice management system which is free, open source, compliant with HIPAA standards and supported on a variety of platforms. The user interface is web based, making the look and feel more familiar to less technical users and more accessible to all users. Some of the features of OpenEMR include management of electronic health records, electronic billing system, support for multiple coding systems, prescriptions, patient statements and reporting. OpenEMR was built upon

LAMP (an acronym for Linux, Apache, MySql and Php/Perl/Python) architectural platform. It provides the basic role-based access control, which protects data in the system from inappropriate accesses. Our system enhances the access control mechanism by incorporating two important components: isolation and delegation. In addition, software modules for the configuration and analysis related with the two components are provided.

3.1   Isolation

Our system provides a dynamic isolation environment for the specified roles to perform their operations inside. Isolation is achieved through dynamically intercepting the database operations by the role and redirecting the operations to the isolated data objects. If there is no isolated data object originally inside the isolation environment, a copy-on-write operation will be invoked to prepare such an isolated copy. This can be implemented at three different levels: database level, table level and record level. We chose the table level approach for the sake of simplicity and reasonable performance and storage overhead. With that, there will be two versions of the same table in our system after the isolated roles modify the table, for instance, *patient_data* table is the original table for storing the patient records, and *ipatient_data* table is its counterpart inside the isolation environment. At the end of the isolated session, a summary log of conflicts or violations will be automatically created by verifying all the isolated operations against the security and consistency policies specified by the system administrators.

We provided an isolation administration web page as shown in Figure 2. This page is visible only to the administrator. And the administrator can add/remove isolated roles and specify isolated users. He can also view the log information of each isolated user from the log column.
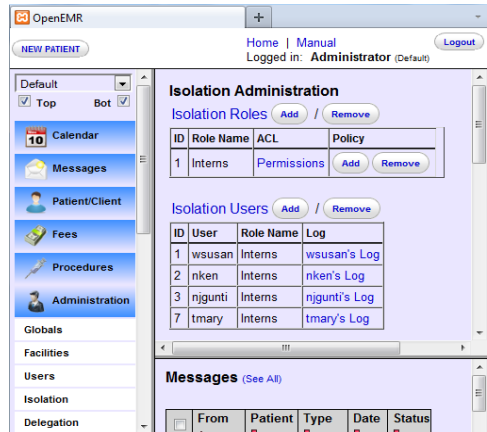
Fig. 2. Isolation Administration

3.2   Delegation

The delegation component is implemented by developing a delegation server which decides to grant the delegation requests based on certain rules. There are two delegation modes, active delegation and passive delegation, which are used for emergency situations and normal cases, respectively. One unique feature of our system is that isolation is used to support the delegation service. Even if the delegation request is granted, the system can isolate the operations of the delegatee, and then verify the net-effect of the isolated operations before applying them to the original system. This provides another level of protection because it is possible that the delegatee could

commit unexpected errors. In addition, the administrator has a delegation administration web page to manage delegation related functionalities.

We evaluated our system by defining the intern doctors as the special role to be isolated. Instead of being denied any access to the system, the intern doctors could access the system inside an isolation environment during the training phase. They could also request the active delegation under emergency situations. The accumulated isolated operations would be automatically verified against the security/consistency policies after the session of the intern doctors and the violations would be reported to the system administrators.

4   Conclusions and Future Work

Our system can grant the needed accesses to medical professionals, such as intern doctors, in a secure and controlled manner. Those accesses would be otherwise denied in HISs with traditional access control models. In our system, the isolation and delegation services can be configured to support access requests from certain special roles (or users). For the future work, we plan to realize the isolation environment in a more efficient way by implementing copy-on-write and redirection operations at the record level. We will also enhance the delegation service by incorporating the trustworthiness and capability values of the delegation candidates. In addition, we plan to invite medical professionals to use and evaluate our system, and their feedback will be analyzed and used to further improve our system.

References

1. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Trans. on Information and System Security (TISSEC), 224–274 (August 2001)
2. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-Based Access Control Models. Proc. IEEE Computer Security 29(2), 38–47 (1996)
3. Duh, C.J.: EMR's Will Save Time and Improve Coordination. From Doctors for America, `http://www.drsforamerica.org/blog/emr-s-will-save-time-and-improve-coordination` (retrieved August 22, 2011)
4. Gunti, N., Sun, W., Niamat, M.: I-RBAC: Isolation Enabled Role-Based Access Control. In: 9th Annual Conference on Privacy, Security and Trust, Quebec, Canada, pp. 79–86. IEEE Computer Security Press (2011)
5. Na, S., Cheon, S.: Role Delegation in Role-Based Access Control. In: 5th ACM Workshop on Role-based Access Control, Berlin, Germany, pp. 39–44. ACM Press (2000)

Author Index