Yunjun Gao Kyuseok Shim
Zhiming Ding Peiquan Jin
Zujie Ren Yingyuan Xiao
An Liu Shaojie Qiao (Eds.)

LNCS 7901

Web-Age Information Management

WAIM 2013 International Workshops: HardBD, MDSP, BigEM, TMSN, LQPM, BDMS Beidaihe, China, June 2013, Proceedings

Springer

Lecture Notes in Computer Science 7901

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

Yunjun Gao Kyuseok Shim Zhiming Ding
Peiquan Jin Zujie Ren Yingyuan Xiao
An Liu Shaojie Qiao (Eds.)

Web-Age Information Management

WAIM 2013 International Workshops:
HardBD, MDSP, BigEM, TMSN, LQPM, BDMS
Beidaihe, China, June 14-16, 2013
Proceedings

Springer

Volume Editors

Yunjun Gao, Zhejiang University, Hangzhou, China
E-mail: gaoyj@zju.edu.cn

Kyuseok Shim, Seoul National University, Korea
E-mail: shim@ee.snu.ac.kr

Zhiming Ding, Chinese Academy of Sciences, Beijing, China
E-mail: zhiming@iscas.ac.cn

Peiquan Jin, University of Science and Technology of China, Hefei, China
E-mail: jpq@ustc.edu.cn

Zujie Ren, Hangzhou Dianzi University, China
E-mail: renzju@gmail.com

Yingyuan Xiao, Tianjin University of Technology, China
E-mail: yyxiao@tjut.edu.cn

An Liu, University of Science and Technology of China, Hefei, China
E-mail: liuan@ustc.edu

Shaojie Qiao, Southwest Jiaotong University, Chengdu, China
E-mail: sjqiao@swjtu.edu.cn

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-39526-0 e-ISBN 978-3-642-39527-7
DOI 10.1007/978-3-642-39527-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013942401

CR Subject Classification (1998): H.3, H.4, H.2.8, H.2.4, C.2.1, E.1, F.2.2

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Web-Age Information Management (WAIM) is a leading international conference for researchers, practitioners, developers, and users to share and exchange their cutting-edge ideas, results, experiences, techniques, and tools in connection with all aspects of Web data management. The conference invites original research papers on the theory, design, and implementation of Web-based information systems. As the 14th event in the increasingly popular series, WAIM 2013 was held in Beidaihe, China, during June 14–16, 2013.

Along with the main conference, WAIM workshops intend to provide international groups of researchers with a forum for the discussion and exchange of research results contributing to the main themes of the WAIM conference. This WAIM 2013 workshop volume contains the papers accepted for the following six workshops that were held in conjunction with WAIM 2013. These six workshops were selected after a public call-for-proposals process, each of which focuses on a specific area that contributes to the main themes of the WAIM conference. The six workshops were as follows:

- The International Workshop on Big Data Management on Emerging Hardware (HardBD 2013)
- The Second International Workshop on Massive Data Storage and Processing (MDSP 2013)
- The First International Workshop on Emergency Management in Big Data Age (BigEM 2013)
- The International Workshop on Trajectory Mining in Social Networks (TMSN 2013)
- The First International Workshop on Location-Based Query Processing in Mobile Environments (LQPM 2013)
- The First International Workshop on Big Data Management and Service (BDMS 2013)

All the organizers of the previous WAIM conferences and workshops have made WAIM a valuable trademark, and we are proud to continue their work. We would like to express our thanks and acknowledgments to all the workshop organizers and Program Committee members who contributed to making the workshop program such a success. They put a tremendous amount of effort into soliciting and selecting research papers with a balance of high quality, novelty, and applications. They also followed a rigorous review process. A total of 37 papers were accepted. Last but not least, we are grateful to the main conference organizers and the local Organizing Committee for their great support and wonderful arrangements.

Yunjun Gao
Kyuseok Shim

HardBD 2013 Workshop Organizers' Message

Data properties and hardware characteristics are two key aspects for efficient data management. A clear trend in the first aspect, data properties, is the increasing demand to manage and process Big Data, characterized by the fast evolution of "Big Data Systems," where nearly every aspect of both enterprise and consumer services is being driven by data processing and analysis. Examples of big data systems include NoSQL storage systems, Hadoop/MapReduce, data analytics platforms, search and indexing platforms, and messaging infrastructures. These systems address needs for structured and unstructured data across a wide spectrum of domains such as the Web, social networks, enterprise, cloud, mobile, sensor networks, multimedia/streaming, cyber-physical and high-performance systems, and for multiple application areas such as healthcare, transportation, and scientific computing.

At the same time, hardware characteristics are undergoing rapid changes, imposing new challenges for an efficient utilization of hardware resources. Recent trends include storage-class memory, massive multi-core processing systems, very large main memory systems, fast networking components, big computing clusters, and large data centers that consume massive amounts of energy. It is clear that many aspects of data management need to evolve with these trends. Utilizing new hardware technologies for efficient big data management is of urgent importance.

The First International Workshop on Big Data Management over Emerging Hardware (HardBD 2013) was held on June 14, 2013, at Beidaihe in conjunction with The 14th International Conference on Web-Age Information Management (WAIM 2013). The overall goal of the workshop is to bring together researchers, practitioners, system administrators, system programmers, and others interested in sharing and presenting their perspectives on the effective management of big data over new hardware platforms, and also to discuss and identify future directions and challenges in this area.

The workshop attracted six submissions. All submissions were peer reviewed by at least three Program Committee members to ensure that high-quality papers were selected. On the basis of the reviews, the Program Committee selected three papers for inclusion in the workshop proceedings (acceptation rate 50%). The final program of the workshop also consisted of three invited talks. One of them was from Alibaba, presented by Zhenkun Yang, and the other two were from academia, presented by Changsheng Xie (Huazhong University of Science and Technology) and Nong Xiao (National University of Defense Technology).

The Program Committee of the workshop consisted of 15 experienced researchers and experts. We would like to thank the valuable contributions of all the Program Committee members during the peer review process. Also, we would like to acknowledge the WAIM 2013 Workshop Chairs for their great support of HardBD 2013, and the support from the Natural Science Foundation of China (No.60833005).

Xiaofeng Meng
Theo Härder
Peiquan Jin
Binsheng He

HardBD 2013 Workshop Organization

General Co-chairs

Xiaofeng Meng	Renmin University of China, China
Theo Härder	Technical University of Kaiserslautern, Germany

Program Co-chairs

Peiquan Jin	University of Science and Technology of China, China
Binsheng He	Nanyang Technological University, Singapore

Publicity Chair

Yi Ou	Technical University of Kaiserslautern, Germany

Program Committee

Bin Cui	Peking University, China
Bin He	IBM Almaden Research, USA
Sang-Wook Kim	Hanyang University, Korea
Ioannis Koltsidas	IBM Research - Zurich, Switzerland
Ziyu Lin	Xia'Men University, China
Yi Ou	TU Kaiserslautern, Germany
Ilia Petrov	Reutlingen University, Germany
Vijayan Prabhakaran	Microsoft Research, USA
Jianliang Xu	Hong Kong Baptist University, SAR China

MDSP 2013 Workshop Organizers' Message

On behalf of the Program Chairs for MDSP 2013, consisting of two General Co-chairs and two Program Co-chairs, we are pleased to present you with this volume. It contains the papers accepted for presentation in the workshop program of the 14th International Conference on Web-Age Information Management held in Beidaihe, China, during June 14–16, 2013.

This was the second International Conference on Massive Data Storage and Processing (MDSP). In all, 21 papers were submitted to the MDSP program, from which eight were accepted for presentation and inclusion in the conference proceedings. An acceptance rate of 40% makes MDSP one of the most selective workshops of WAIM 2013.

We would like to thank all the authors of submitted papers for choosing MDSP 2013 for the presentation of their research results. Because of the high quality of the submitted papers, selecting the eight papers for the main conference was a very difficult task. We are deeply indebted to the four program Chairs and 16 Program Committee members for their conscientious and impartial judgment and for the time and effort they contributed in preparation of this year's conference. All Area Chairs and reviewers are listed on the following pages.

The organizers of the conference are very happy with the response to our call for papers, noting the interest of the data storage and processing community in this field. The workshop was composed of eight papers selected for presentation, covering a wide range of topics and showing interesting experiences. A brief summary of all the contributions, classified in three main areas, is presented below.

- "Adaptive Sequential Prefetching for Multiple Streams" submitted by Yong Li, Dan Feng, Zhan Shi, and Qing Liu from Huazhong University of Science and Technology. The authors presented an adaptive sequential prefetching algorithm called ASPM, solving the un-fairness and performance degradation introduced by streams with diverse access rates.
- "Research of Network Coding Data Collection Mechanism Based on The Rough Routing in Wireless Multi-hop Network" submitted by Jian Wan, Ligang He, and Wei Zhang et al. from Hangzhou Dianzi University. The authors proposed a rough routing-based data collection mechanism used in wireless sensor network called BRRCD, which aims to restrain the cliff effect to some extent.
- "Incremental Truth Discovery for Information from Multiple Data Sources submitted by Li Jia, Hongzhi Wang, Jianzhong Li" and Hong Gao from Harbin Institute of Technology. The authors presented an incremental strategy for discovering truth in multisource integration using boosting-like ensemble classifiers.

- "Simdedup: A New Deduplication Scheme Based on Simhash" submitted by Wenbin Yao and Pengdi Ye from Beijing University of Posts and Telecommunications. The authors presented a near-exact deduplication scheme named Simdedup, which exploits file similarity and chunk locality to improve the accuracy of deduplication.
- "InfoMall: A Large-Scale Storage System for Web Archiving" submitted by Lian'en Huang, Jinping Li, and Xiaoming Li from Peking University Shenzhen Graduate School. The authors proposed a system designed for storing massive Web pages effectively and efficiently.
- "Sequential Record Based Compression for Massive Image Storage in Database1" submitted by Ziyun Ma, Xiaonian Wang, and Ping Jiang et al. from Tongji University. The authors proposed an image compression scheme that is learnt from video compression to remove temporal and spatial redundancy in the image sequence.
- "Continuous, Online Anomaly Region Detection and Tracking in Networks" submitted by Shuiyuan Xie, Xiuli Ma, and Shiwei Tang from Peking University. The authors presented a framework to detect and track an anomaly region continuously.
- "Event Matching Algorithm to Collaborate Sensor Network and the Cloud through a Middleware Service" submitted by Mohammad Hasmat Ullah, Sung-Soon Par, and Gyeong Hun Kim from the Department of Computer Science and Engineering, Anyang University, South Korea and Gluesys Co., Ltd. The authors proposed a content-based event-matching algorithm to analyze subscriptions and match proper content easily to convey WSN-driven data to the subscribers.

We would like to thank everyone who helped us. We greatly appreciate the advice and support by the WAIM 2013 General Co-chairs, Xiaofeng Meng (Renmin University of China) and Huan Liu (Arizona State University, USA), Program Co-chairs, Jianyong Wang (Tsinghua University, China) and Hui Xiong (Rutgers University, USA), and Workshops Chairs Yunjun Gao(Zhejiang University) and Kyuseok Shim (Seoul National University, South Korea).

Weisong Shi
Yunjun Gao
Weiping Wan
Zujie Ren

MDSP 2013 Workshop Organization

General Co-chairs

Weisong Shi	Wayne State University, USA
Yunjun Gao	Zhejiang University, China

Program Co-chairs

Weiping Wang	Chinese Academy of Sciences, China
Zujie Ren	Hangzhou Dianzi University, China

Program Committee

Guoray Cai	Pennsylvania State University, USA
Yong Woo Lee	University of Seoul, Korea
Hung Keng Pung	National University of Singapore, Singapore
Xiaofei Liao	Huazhong University of Science and Technology, China
Yijun Bei	Zhejiang University, China
Yi Zhuang	Zhejiang GongShang University, China
Tao Jiang	Jiaxing University, China
Weiwei Sun	Fudan University, China
Xiaokui Xiao	Nanyang Technological University, Singapore
Yimin Lin	Singapore Management University, Singapore
Bin Yao	Shanghai Jiaotong University, China
Shaojie Qiao	Southwest Jiaotong University, China
Congfeng Jiang	Hangzhou Dianzi University, China
Jilin Zhang	Hangzhou Dianzi University, China
Qi Qiang	Alibaba Corp., China
Yongjian Ren	Infocore Corp., China

BigEM 2013 Workshop Organizers' Message

With the advances in emergency management and information communication technologies, improving the efficiency and accuracy of emergency management systems through modern data processing techniques has become a crucial research issue. The past decade has witnessed huge technical advances in sensor networks, Internet/Web of Things, cloud computing, mobile/embedded computing, spatial/temporal data processing, and big data, and these technologies have provided new opportunities and solutions to emergency management. Data properties and hardware characteristics are two key aspects for efficient data management.

Data processing in emergency management is a typical big data scenario. Numerous sensors and monitoring devices continuously sample the states of the physical world, while the Web data processing techniques make the Internet a huge data repository that can reflect the states of the cyber world and the human world. The efficient processing of these data imposes a big challenge to the data management community. It is important to develop advanced data management and data processing mechanisms to support disaster detection, disaster response and control, rescue resource planning and scheduling, and emergency commanding.

The First International Workshop on Emergency Management in Big Data Age (BigEM 2013) was held in conjunction with the 14th International Conference on Web-Age Information Management (WAIM 2013) in Beidaihe, China, during June 14, 2013. The purpose of BigEM 2013 was to provide a forum for researchers and practitioners to exchange ideas and progress in the related areas of big data management, such as cloud computing, parallel algorithms, internet of things, spatial database, complex event detection, optimization theory, intelligent transportation systems, and social networks. The workshop attracted eight submissions. All submissions were reviewed by at least three Program Committee members in order to ensure that high-quality papers were selected. Following a rigorous review process, eight papers were selected for publication in the workshop proceedings.

The Program Committee of BigEM 2013 consisted of 18 experienced researchers and experts in the area of big data management. We would like to thank all the authors for submitting their papers to the workshop and the valuable

contributions of all the Program Committee members during the peer-review process. Also, we would like to thank the WAIM 2013 Workshop Co-chairs for their great support in ensuring the success of BigEM 2013.

Feiyue Wang
Xiaofeng Meng
Zhiming Ding
Dajun Zeng
Jinzhou Li

BigEM 2013 Workshop Organization

General Co-chairs

Feiyue Wang	Chinese Academy of Sciences, China
Xiaofeng Meng	Renmin University of China, China

Program Co-chairs

Zhiming Ding	Chinese Academy of Sciences, China
Dajun Zeng	Chinese Academy of Sciences, China
Jianhui Li	Chinese Academy of Sciences, China

Program Committee

Xiaofang Zhou	University of Queensland, Australia
Hui Zhang	Tsinghua University, China
Xiaogang Qiu	National University of Defense Technology, China
Lifeng Zhang	Renmin University of China, China
Kian-Lee Tan	National University of Singapore, Singapore
Panos Kalnis	KAUST, Saudi Arabia
Zhidong Cao	Chinese Academy of Sciences, China
Yi Yang	Carnegie Mellon University, USA
Rui Yang	Tsinghua University, China
Hong Huang	Tsinghua University, China
Hua Lu	Aalborg University, Denmark
Wei Xu	Renmin University of China, China
Jiajie Xu	Chinese Adademy of Sciences, China
Chengfei Liu	Swinburne University of Techonology, Australia
Julien Bourgeois	University of Franche-Comte, France
Kuien Liu	Chinese Adademy of Sciences, China
Lei Chen	Hong Kong University of Science and Technology, SAR China
Michael Marschollek	University of Braunschweig, Germany

TMSN 2013 Workshop Organizers' Message

Social networks and social media are prevalent on the Internet and have become an active research topic attracting many professionals and researchers from a variety of fields. By adding trajectory information, we can bring online social networks and media back to the physical world and share our real-life experiences in the virtual world conveniently. By mining trajectory patterns or predicting locations from social networks, people can not only track and share location-related information with each other via mobile devices or desktop computers, but can also leverage collaborative social knowledge learned from user-generated and location-related content.

As trajectory is one of the most important properties in people's everyday lives, the research on trajectory mining in social networks (TMSN) bridges the gap between online societies and the physical world, and enables novel applications that have the potential to change the way we live, e.g., path planning/prediction, friend suggestion, location/friend recommendations, community discovery, human mobility modeling, and user activity analysis. The technology derived from TMSN, e.g., trajectory mining and retrieval, can be applied to a multitude of other research areas including biology, sociology, geography, and climatology.

The International Workshop on Trajectory Mining in Social Networks (TMSN 2013), co-located with the 14th International Conference on Web-Age Information Management (WAIM 2013), was held in Beidaihe, China, during June 14, 2013. The goal of TMSN 2013 was to broaden the focus to include location-based social media more generally, and bring together researchers and practitioners from academia and industry to discuss and share the state of the art in TMSN development and application, present their ideas and contributions, and set the future direction of TMSN research. The workshop attracted three submissions. All submissions were reviewed by at least three Program Committee members in order to ensure that high-quality papers were selected. Following a rigorous review process, three papers were selected for publication in the workshop proceedings.

The Program Committee of TMSN 2013 consisted of 20 experienced researchers and experts in the area of data management. We would like to thank all the authors for submitting their papers to the workshop and the valuable

contributions of all the Program Committee members during the peer-review process. Also, we would like to thank the WAIM 2013 Workshop Co-chairs for their great support in ensuring the success of TMSN 2013.

Yu Zheng
Yunjun Gao
Shaojie Qiao
Cheqing Jin

TMSN 2013 Workshop Organization

General Co-chairs

Yu Zheng	Microsoft Research Asia, China
Yunjun Gao	Zhejiang University, China

Program Co-chairs

Shaojie Qiao	Southwest Jiaotong University, China
Cheqing Jin	East China Normal University, China

Program Committee

Xiaofang Zhou	University of Queensland, Australia
Weining Qian	East China Normal University, China
Cheqing Jin	East China Normal University, China
Jiancheng Lv	Sichuan University, China
Daniel D. Zeng	The University of Arizona, USA
Christopher C. Yang	Drexel University, USA
Warren H. Jin	CSIRO Mathematics, Informatics and Statistics, Australia
Michael Chau	The University of Hong Kong, SAR China
Lei Zou	Beijing University, China
Hongzhi Wang	Harbin Institute of Technology, China
Yu Gu	Northeastern University, China
Zhaonian Zou	Harbin Institute of Technology, China
Tiaocheng Zhang	Northeastern University, China
Jianbin Huang	Xidian University, China
Yu Cao	EMC Information Technology R & D, China
Lu-an Tang	University of Illinois, Urbana Champaign, USA
Dongxiang Zhang	National University of Singapore, Singapore
Liangxu Liu	Ningbo University of Technology, China
Zujie Ren	Hangzhou Dianzi University, China
Jiaoling Zheng	Chengdu University of Information Technology, China
Ning Yang	Sichuan University, China

LQPM 2013 Workshop Organizers' Message

Geographical mobility has become a highly desirable aspect of today's information processing paradigm. The rapid advances in electronics and wireless communication make it possible for mobile users to access pervasive information from anywhere at anytime. Now, more and more mobile devices with computing and wireless communication capabilities are being widely used. The majority of these mobile devices are equipped with positioning systems, which gave rise to a new class of mobile services, i.e., location-based services (LBS). LBS enable users of mobile devices to search for facilities such as restaurants, shops, and car-parks close to their route. Owing to the ever-growing popularity of mobile devices and the sharply increasing query loads, to efficiently answer location-based queries in mobile computing environments is becoming more and more challenging. This requires innovative location-based query processing technologies and scalable processing system.

The First International Workshop on Location-based Query Processing in Mobile Environments (LQPM 2013) was held in conjunction with the 14th International Conference on Web-Age Information Management (WAIM 2013) in Beidaihe, China, during June 14, 2013. LQPM 2013 intended to bring together researchers and practitioners from academia and industry to discuss cutting-edge research on location-based query processing in mobile environments, such that the interactions among the experts of these areas can be promoted and new interdisciplinary technologies can be developed. The scope of the workshop includes theories, techniques, and novel applications involving location-aware query processing in mobile environments. The workshop attracted six submissions. All submissions were reviewed by at least three Program Committee members in order to ensure that high-quality papers were selected. Following a rigorous review process, six papers were selected for publication in the workshop proceedings.

The Program Committee of LQPM 2013 consisted of 20 experienced researchers and experts in the area of data management. We would like to thank all the authors for submitting their papers to the workshop and the valuable contributions of all the Program Committee members during the peer-review process. Also, we would like to thank the WAIM 2013 Workshop Co-chairs for their great support in ensuring the success of LQPM 2013.

Xiaojie Yuan
LihChyun Shu
Hongya Wang
Yingyuan Xiao

LQPM 2013 Workshop Organization

General Co-chairs

Xiaojie Yuan	Nankai University, China
LihChyun Shu	National Cheng Kung University, Taiwan

Program Co-chairs

Hongya Wang	Donghua University, China
Yingyuan Xiao	Tianjin University of Technology, China

Program Committee

LihChyun Shu	National Cheng Kung University, Taiwan
Xiaojie Yuan	Nankai University, China
Yingyuan Xiao	Tianjin University of Technology, China
Hongya Wang	Donghua University, China
Kevin Lu	Brunel University, UK
Weiwei Sun	Fudan University, China
Cheqing Jin	East China Normal University, China
Guohua Liu	Donghua University, China
Depeng Dang	Beijing Normal University, China
Shijun Li	Wuhan University, China
Biao Qin	Renmin University of China, China
Jinguang Gu	Wuhan University of Science and Technology, China
Lizhen Cui	Shandong University, China
Guoqiong Liao	Jiangxi University of Finance and Economics, China
Pingpeng Yuan	Huazhong University of Science and Technology, China
Yukun Li	Tianjin University of Technology, China
Xin Wang	Tianjin University, China
R. Padmavathy	National Institute of Technology, India
Xiaolong Zhang	Netease Corp, China
Jia Xu	Northeastern University, China

BDMS 2013 Workshop Organizers' Message

The proliferation of new technologies such as the Internet of things, social networks, and cloud computing produces datasets whose volume, velocity, and variability is beyond the ability of commonly used software tools. Efficient processing of big data creates notable research challenges and opportunities for traditional databases and infrastructure. On the other hand, computing and software are turned into commodity services by cloud computing, so it is necessary to examine big data as a service (BDaaS) from both technology and business aspects.

The First International Workshop on Big Data Management and Service (BDMS 2013) was held in conjunction with the 14th International Conference on Web-Age Information Management (WAIM 2013) in Beidaihe, China, during June 14, 2013. The purpose of BDMS 2013 was to bring together researchers, practitioners, and developers from academia and industry to discuss cutting-edge research on big data management and service, such that the interactions among the experts of these areas can be promoted and new interdisciplinary technologies can be developed. The scope of the workshop includes big data management techniques such as big data modeling, analytics, and toolkits, as well as techniques and case studies for providing big data as a service. The workshop attracted six submissions. All submissions were reviewed by at least three Program Committee members in order to ensure that high-quality papers were selected. Following a rigorous review process, six papers were selected for publication in the workshop proceedings.

The Program Committee of BDMS 2013 consisted of 13 experienced researchers and experts in the area of data management. We would like to thank all the authors for submitting their papers to the workshop and the valuable contributions of all the Program Committee members during the peer-review process. Also, we would like to thank the WAIM 2013 Workshop Co-chairs for their great support in ensuring the success of BDMS 2013.

Zhe Shan
An Liu
Xiaoling Wang
Liwei Wang
Kai Zheng

BDMS 2013 Workshop Organization

General Co-chairs

Zhe Shan	Manhattan College, USA
An Liu	University of Science and Technology of China, China

Program Co-chairs

Xiaoling Wang	East China Normal University, China
Liwei Wang	Wuhan University, China
Kai Zheng	The University of Queensland, Australia

Program Committee

Wei Chen	Beijing Institute of Technology, China
Cheqing Jin	East China Normal University, China
Qizhi Liu	Nanjing University, China
Bin Mu	Tongji University, China
Xuequn Shang	Northwestern Polytechnical University, China
Weilai Sun	Fudan University, China
Mingjun Xiao	University of Science and Technology of China, China
Haoran Xie	City University of Hong Kong, SAR China
Lizhen Xu	Southeast University, China
Xiaochun Yang	Northeastern University, China
Kun Yue	Yunnan University, China
Dell Zhang	University of London, UK
Xiao Zhang	Renmin University of China, China

Table of Contents

The International Workshop on Big Data Management on Emerging Hardware (HardBD 2013)

The Second International Workshop on Massive Data Storage and Processing (MDSP 2013)

The First International Workshop on Emergency Management in Big Data Age (BigEM 2013)

The International Workshop on Trajectory Mining in Social Networks (TMSN 2013)

The First International Workshop on Location-Based Query Processing in Mobile Environments (LQPM 2013)

The First International Workshop on Big Data Management and Service (BDMS 2013)

Power Conservation in Large-Scale Storage Systems

Qiang Cao and Changsheng Xie

Data Storage Division, Wuhan National Laboratory for Optoelectronics,
Huazhong University of Science and Technology, Wuhan, China

Abstract. The amount of data in large scale storage systems in data centers has been exponentially increasing for the last decade. The necessity for capping power consumption has significantly restricted the potential of modern data centers. Consequently, energy conservation techniques for storage systems are gaining growing popularity to tackle the challenge of energy consumption problem. In this talk, we will present an overview of our project of power conservation techniques in large-scale storage systems, which was launched in 2010 and supported by the National Natural Science Foundation of China (No. 60933002). The project focuses on constructing fundamental theories, designing and implementing low-energy-consumption storage systems, and energy-efficient approaches including measure mechanism, low-power storage media and device, server architecture, tradeoff between performance and energy, schedule algorithms, workload pattern and large scale system, etc. Firstly, we implemented a new integrated framework called TRACER for large scale storage systems [1]. The TRACER is a load-controllable energy-efficiency evaluation framework, which facilitates a trace replay mechanism for mass storage systems. TRACER consists of performance and energy metrics as well as a toolkit used to measure energy efficiency of storage systems. Using the measure tool, we further evaluated how hardware and software configurations impact energy consumption and performance under four typical storage workloads generated by the benchmarks such as fileserver, vermeil, webserver and OLTP [2]. Inspired by the practical observations, we will discuss some principles for power conservation in mass storage system. Besides, during the past three years, we have made some achievements in some new high energy efficient RAID scheme and fast recovery mechanisms, such as DROP [3], PERAID [4], VDF [5] and SPA [9]. For large scale storage systems, some new scalable redundancy modes have been presented, such as HDP code [6], H-code [8] and Code-M [10]. Furthermore, we also focus on new low power storage media as SSD and its implementation [9]. After a brief introduction of the current status of the project, some experiences and lessons concluded from the study will be discussed. Finally, we will introduce several open issues and potential directions of the research within the scope of power conservation for large-scale storage systems.

Keywords: Power conservation, large-scale storage systems, RAID.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 1–2, 2013.

References

1. Liu, Z., Wu, F., Qin, X., Xie, C., Zhou, J., Wang, J.: TRACER: A Trace Replay Tool to Evaluate Energy-Efficiency of Mass Storage Systems. In: 2010 IEEE International Conference on Cluster Computing (CLUSTER), pp. 68–77 (2010)
2. Liu, C.G., Huang, J.Z., Cao, Q., et al.: Evaluating Energy and Performance for Server-Class Hardware Configurations. In: IEEE NAS 2011, pp. 339–347 (2011)
3. Wan, J., Wang, J., Liu, Y., Yang, Q., Wang, J., Xie, C.: Enhancing Shared RAID Performance Through online Profiling. In: IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST), April 16-20 (2012)
4. Wan, J., Yin, C., Wang, J., Xie, C.: A New High-performance, Energy-efficient Replication Storage System with Reliability Guarantee. In: IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST), April 16-20 (2012)
5. Wan, S., Cao, Q., Huang, J., Li, S., Li, X., Zhan, S., Yu, L., Xie, C.: Victim Disk First: An Asymmetric Cache to Boost the Performance of Disk Arrays under Faulty Conditions. In: USENIX ATC 2011, June 15-17 (2011)
6. Wu, C., He, X., Wu, G., Wan, S., Liu, X., Cao, Q., Xie, C.: HDP Code: A Horizontal-Diagonal Parity Code to Optimize I/O Load Balancing in RAID-6. In: Proceedings of The 41th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2011) (2011)
7. Wu, C., Wan, S., He, X., Cao, Q., Xie, C.: H-Code: A Hybrid MDS Array Code to Optimize Partial Stripe Writes in RAID-6. In: IPDPS 2011, May 19-22 (2011)
8. Tian, L., Cao, Q., Jiang, H., Feng, D., Xie, C., Qin: On-Line Availability Upgrades for Parity-based RAIDs through Supplementary Parity Augmentations. ACM Transactions on Storage 6(4), Article 17 (May 2011)
9. Hu, Y., Jiang, H., Feng, D., Tian, L., Luo, H., Ren, C.: Exploring and Exploiting the Multi-level Parallelism Inside SSDs for Improved. IEEE Transactions on Computers (February 28, 2012)
10. Wan, S., Cao, Q., Xie, C., He, X.: Code-M: a Non-MDS Erasure Code Scheme to Support Fast Recovery from up to Two Disk Failures in Storage System. In: Proceedings of The 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2010 (2010)

A Scalable, Continuously Available Database System Based on Commodity Hardware

Zhenkun Yang

Alibaba, Beijing, China

Abstract. RDBMS is facing several challenges. First, the scalability of RDBMS depends exclusively on single big and highly reliable server which is disproportionately expensive. Second, a few bytes of mutation often causes both read and write of a few kilobytes of data (a page/block) in RDBMS (also known as write amplification) which severely curbs its write transaction performance. Third, due to its mixed read and write infrastructure, it is difficult for RDBMS to take the advantage of modern commodity flash-based SSDs which own perfect random read ability but relatively poor random write performance. In this talk, we will present OceanBase, an open source (https://github.com/alibaba/oceanbase) database system. OceanBase is a distributed system based on commodity hardware. It owns many features of traditional RDBMS, e.g., ACID, key SQL features and SQL interface, etc. as well as a distributed system, e.g., scalability, continuous availability, etc. It also owns a separated read and write architecture which removes the above write amplification in RDBMS and is very friendly to commodity SSDs by eliminating random disk write. It makes the widely used database shading obsolete. There are dozens of OceanBase instances in the production system of Alibaba and serves tens of billions read and write transactions every day. The first instance has been in service for more than 2 years and consists of 80 commodity servers today. One table in the above instances contains more than 100 billion records and a few tens of terabytes of data.

Keywords: Database, Scalability, Continuous Availability, Commodity Hardware.

Reference

1. https://github.com/alibaba/oceanbase

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, p. 3, 2013.

DL-Dedupe: Dual-Level Deduplication Scheme for Flash-Based SSDs

Wanhui He, Nong Xiao, Fang Liu, Zhiguang Chen, and Yinjin Fu

State Key Laboratory of High Performance Computing,
National University of Defense Technology, Changsha, China
{hewanhui,chenzhiguanghit,yinjinfu}@gmail.com,
{nongxiao,fangliu}@nudt.edu.cn

Abstract. Although flash memory-based Solid State Drive (SSD) was born as the replacement of Hard Disk Drive (HDD) due to its high performance and low power consumption, the limited write endurance of flash and the reliability problem hold back the footsteps of SSD popularization. We propose DL-Dedupe, a dual-level deduplication scheme for flash-based SSDs to reduce the amount of data written to flash memory and enhance the storage capacity at the same time by eliminating unnecessary replicated write operations, further reducing garbage collection to improve the lifespan of SSD. It combines chunk and page level deduplication to save memory utilization, reduce access latency and ensure deduplication effectiveness. It also introduces hybrid address mapping and adaptive cache replacement scheme to accelerate index lookup and reduce memory overhead for desirable access speed. The experimental results show that DL-Dedupe can significantly reduce redundant writes, thus enhance the reliability and lifespan of SSD.

Keywords: SSD, deduplication, lifespan, dual-level.

1 Introduction

With the continuous technological improvement in the last few decades, flash memory is developing at a rapid speed. The density of flash memory is increasing, while the price is decreasing. Flash-based SSD is widely used in personal computers, mobile phones, MP3 and other small electronic devices due to its fast access speed, low power consumption and stable performance [1]. However, the limited lifespan is always the Achilles' heel that causes the reliability problem of flash-based SSD. Besides lifespan, the limited capacity is another obstacle for the popularization of SSD.

Data deduplication, an effective data compression approach that exploits data redundancy, partitions large data objects into smaller parts and replaces these chunks by their fingerprints for the purpose of communication or storage efficiency [2], has received comprehensive interest recently. It becomes more and more important due to the rapid growth of amount of enterprise data, the highly redundant dataset and the continuously increasing data transmission rate. No doubt that data deduplication has become a necessary part of current big data storage. As the volume of data stored in

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 4–15, 2013.

datacenter is up to PB or even EB level, large amounts of data detection and comparison as well as the metadata storage will consume massive computational and memory resources. At the same time, disk bottleneck may impose serious impact on the storage system performance.

Both flash-based Solid State Drives and data deduplication have been the hotspots of storage research area for a long time. However, both of them have their own fatal weakness. Researchers figure out the combination of them that can foster strengths and circumvent weaknesses. On one hand, the disk bottleneck caused by searching operations in deduplication will be alleviated by introducing SSDs which have much faster read and write speeds than HDDs. On the other hand, the Achilles' heel of flash would be hidden when deduplication is deployed, because less redundant writes are issued to flash and the lifespan of SSD can be significantly enhanced.

Based on the analysis above, we proposed DL-Dedupe, a dual-level deduplication scheme for flash-based SSDs, to enhance the lifespan of SSD by effectively reducing the amount of data written to flash. Moreover, it will reduce the number of garbage collections thus further reduce accesses to flash and improve reliability of SSD.

The contributions of this paper include following aspects:

- We propose a dual-level deduplication scheme for flash-based SSDs, which deploys deduplication technology to improve the reliability and capacity of SSD.
- Neither chunk-level nor page-level deduplication alone can be deployed to achieve desired performance. We attempt to integrate them to pursue effective deduplication and low resource consumption. Furthermore, we also conduct sensitivity study on DL-dedupe granularity in our evaluation to achieve optimal performance with suitable chunk size.
- The DRAM of SSD is not enough to keep all of the deduplication metadata because of the growing of data, so that the metadata will be stored in flash to avoid memory bottleneck. We maintain the frequently accessed fingerprints in DRAM. When it is failed to get the fingerprint in DRAM, an access to flash must be issued. Part of our work is to figure out the appropriate size of the cache and design the effective method for quick search in DRAM.
- As DRAM has been occupied by more and more metadata, the cache replacement scheme is critical for the performance of flash-based SSDs. DL-Dedupe takes both chunk and page fingerprint into account, proposing an adaptive cache replacement scheme, which will reduce the access latency significantly.
- Lastly, we demonstrate that, our prototype of DL-Dedupe can significantly save storage space and improve the performance over the state-of-the-art page deduplication policy for flash-based SSDs via trace-driven experimental evaluation.

2 Background and Motivation

2.1 Background

The processing speed of CPU is doubled every 18 months according to the Moore's Law. However, the storage speed becomes the bottleneck of the system because it has

fallen far behind the development of the processor. Fortunately, the appearance of SSD can effectively alleviate the bottleneck of the system.

At the same time, the breakout of data results in insufficient storage volume and high cost of data management. However, International Data Corporation (IDC) reported that 75% of the data is duplicated, and the redundancy is growing with the passage of time [3]. Data deduplication[4,5,6] has come out to effectively solve these problems. It is good at deleting redundant data to significantly save storage space and network bandwidth, thus minimizing power consumption and the cost. It is not always possible to keep all of fingerprints of deduplication system in memory because of the uncontrollable growth of data. When storing them in the disk, disk bottleneck may become the critical problem. In order to alleviate disk bottleneck, some researchers show interests in flash memory.

Compared to HDD, SSD performs better in many aspects, such as random access speed, low power consumption and high reliability. Because it has no moving parts or mechanical wear-out, it is silent and resistant to heat and shock [1]. According to statistics of market research institute IHS iSuppli [7], SSD can be expected annual growth of 53% in 2013, and the output value will be doubled-digit growth every year in the next three years with the decline in the cost of NAND flash. However, the limited lifetime of SSD is a major drawback that hinders their deployment in reliability sensitive environments. Even worse, it is irreparable if damaged.

The weak point of SSD lies in three main aspects [1].1) Flash page must be erased before it can be written, and the erase/program cycles are limited, that 10,000 times for MLC and 100,000 times for SLC. 2) The out-of-place update feature may result in more invalid pages and further induce garbage collection. 3) Flash memory writes in granularity of pages while erases in granularity of blocks. This characteristic requires transferring the valid pages to vacant blocks for garbage collection, introducing more writes and further aggravates the lifespan problem of SSD.

Quantities of previous research have tries to solve this problem [8,9], including wear-leveling with the spare space of SSD, presence of spatial locality with a write buffer[10] and improving garbage collection policy. In this paper, we propose DL-Dedupe, a dual-level deduplication policy for flash-based SSDs to improve the lifespan of SSD in another way. DL-Dedupe can effectively reduce written data to flash by chunk and page level deduplication in SSD controller, with hybrid mapping policy and adaptive cache replacement to further improve the search speed of fingerprints.

2.2 Motivation

Besides the fact that deploying deduplication in SSD can foster strengths and circumvent weaknesses, existing studies also note that there is a large amount of redundant data existing in the file system, including new version of backup files or even totally identical files. In this section, we investigate the nature of content similarity and access in file system to guide our research using three workloads.

Characteristics of adopted traces are listed in Table 1. They come from three active production systems of Florida International University (FIU) Computer Science department, including *web-vm*, *mail* and *homes*. We downloaded the workload from

their trace website [11], which involves three-week workloads of a virtual machine running two web-servers, an email server, and a file server. According to the introduction [12, 13], *homes* is collected in a NFS server that serves the home directories of their small-sized research group's activities. *web-vm* is collected from a virtualized system that hosts two CS department web-servers, one hosting the department's online course management system and the other hosting the department's web-based email access portal. At last, *mail* is collected from the e-mail server containing similar mailing-list emails and circulated attachments across user INBOXes.

Because of the intractable size of workloads, we just use one week of them to do our experiments. Total, Write and Redundant in Table 1 represent the number of total, write and redundant requests in the workload separately. We can see from the write ratio of W/T in Table 1 that these workloads are write intensive, especially *homes*, whose write operations take 97% of all the requests. So that it is critical to improve write performance. The data redundancy rate, indicated as R/T in Table 1 is relatively high, especially 90% in *mail*. Minority of the dataset is accessed and much storage space can be saved if the redundant data are deleted. Under this observation, we explored DL-Dedupe, which can significantly reduce the redundant writes to flash through a dual-level deduplication policy, thus enhancing the lifespan and expanding the volume of SSD.

Table 1. Characteristics of trace

Traces	Size(GB)	Total	Write	Redundant	W/T(%)	R/T(%)
homes	3.73	12140668	11769606	8262249	97	70
web-vm	1.15	3881712	2989409	1723420	77	58
mail	11.7	15537531	137759031	123673949	89	90

3 Design and Implementation

In order to reduce the amount of data written to flash, we deploy a deduplication module in DL-Dedupe besides address mapping, garbage collection and wear leveling. It is designed in dual level because both memory consumption and access latency will be reduced with the advantages of both page and chunk level deduplication. Moreover, it may further reduce garbage collection and improve the throughput of SSD. In DL-Dedupe, all of the metadata stored in DRAM must be transferred to the specialized area of flash by a flush command before the system is powered off. When it is powered on, the metadata such as mapping tables and part of fingerprints will be restored in DRAM, ensuring the system to work properly.

3.1 An Architectural Overview

The architectural overview of DL-Dedupe is illustrated in Fig.1. It indicates the process of handling a write request in DL-Dedupe. (1) When a write request is issued

to SSD from Block I/O interface, it will be divided into chunks. (2) By applying a hash function to each chunk, we obtain a hash value which is called fingerprint. The fingerprint is the unique identification of the chunk. (3) When the chunk fingerprint hits in DRAM, the chunk address mapping table will be modified and the related write request to flash will be cancelled. (4) If the chunk fingerprint is missed in DRAM, the fingerprint will be added to chunk fingerprint table in DRAM and the chunk will be divided into pages further. (5)The fingerprint of every page will be calculated and looked up in DRAM, if a match is found, we update the page address mapping table and cancel the page written to flash. (6)Otherwise, a lookup in flash will be taken, and the whole flash page the searched fingerprint involved in must be added to DRAM. (7)In the worst case, if all of the lookups end up in failures, the write will be performed to the flash memory as a regular write.

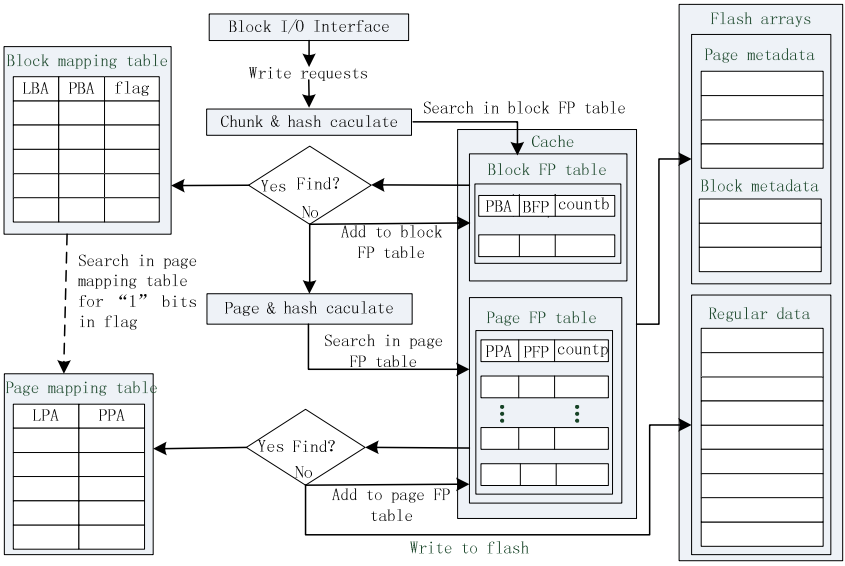


Fig. 1. DL-Dedupe architecture

We set chunk size certain multiples of the flash page size instead of the flash block. It is finally determined through experimental validation, so that we can achieve the optimum performance with suitable chunk size. When data updates, the count number of corresponding shared physical address will be altered and a new shared physical address is allocated. Because of the hybrid address mapping policy and the existence of different logical addresses with shared physical address, the garbage collection is triggered only when no corresponding logical address shares any of the pages in the block. Since the hash value calculation and index lookup may affect the performance, we focus on how to reduce the access latency, including reforming address mapping policy and fingerprint storage management.

3.2 Hybrid-Address Mapping

The DL-Dedupe employs a hybrid address mapping policy. When a write request is issued to the SSD, the incomplete block mapping table will be searched and updated first. If a page update involves in it, then the page mapping table will be modified. The so-called incomplete block mapping table is based on the size of chunk instead of the flash block. This optimization helps to promote the performance under different workloads. The entry of block mapping table involves a flag of N-bit, specifying which page of the chunk has been updated, and N means the number of pages in a chunk.

Solid State Drives are compatible to traditional disk file system because of the flash translation layer. Different from the smallest unit of 4KB flash write, requests handled by block I/O layer are divided into 512-Byte sectors. FTL has to arrange these sectors into 4KB pages and write into flash. Assuming that a block in flash contains 64 pages, the logical address of the write request issued to SSD is just like what is shown in Fig. 2. The lowest 3 bits are always 000, which means writing 8 sectors, actually 4KB, to flash at one time. 6 represents 64 pages in each block and 23 indicates a logical block address. Combination of 23 and 6 represents a logical page address in the page mapping table. Fig.2 also illustrates how the mapping table is searched when a page in the stored block has been updated. The second bit of logic 0 in block mapping is "1", which means the corresponding page has modified and should be searched in the page mapping table. The corresponding logical page address is determined by logical address of modified block offsets 6 bits and the modified page offset in the block. And it is mapped to a physical page address with new block number and page offset.

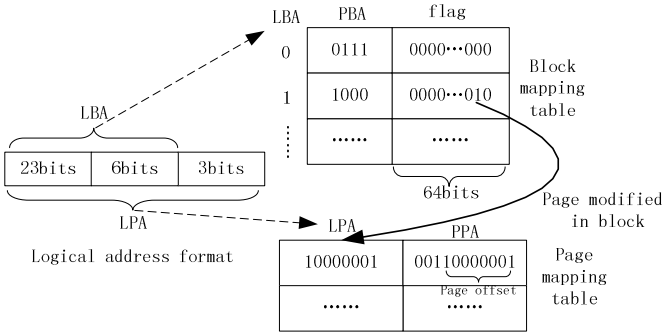


Fig. 2. Hybrid address mapping. When the second page in the logic block 0 is modified, it will be searched in the page mapping table.

3.3 Fingerprint Storage and Management

The metadata of DL-Dedupe increases with the growing amount of the data in storage system. Metadata usually stores in the out-of-region OOB. However, it must be stored to be looked up for indexing in DL-Dedupe, and the shared physical address may be

related to several logical addresses. So, we keep the metadata in specific storage area of SSD in this work.

To take advantages of the high performance of DRAM and the spatial locality of data stream, we also keep frequently accessed fingerprints in DRAM. The DRAM space used to keep fingerprints is divided into chunk and page area, whose capacities are adaptively allocated according to different workloads. Infrequently accessed fingerprints will be replaced out of the allocated DRAM space called cache by the adaptive replacement policy when there is no spare room. When a fingerprint is replaced out of the cache, it will be first kept in the write buffer in DRAM, which has the same size as a flash page. When the write buffer is full, fingerprints in the buffer are flushed to SSD. This optimization helps to avoid fragments in flash.

A chunk fingerprint can efficiently delete a redundant chunk, but it hits in cache not as frequent as a page fingerprint. Page fingerprints, on the other hand, are not so efficient in size, but hit frequently. Considering that it is necessary to keep chunk and page fingerprint indexes in cache of DL-Dedupe, we figure out an adaptive replacement policy to dynamically balance the DRAM capacity allocated to them, thus achieving high hit ratio.

The policy is described as follows:

1. When the system is powered on, the cache is divided equally into L1 and L2.Two LRU arrays are built up in cache to separately keep chunk and page fingerprints.
2. We define P1 for the volume of L1 and P2 for L2. When a chunk fingerprint hits in L1, P=P+delta. Delta is determined by the size of L1 and L2, making P added step by step.
3. If it misses in L1, the chunk will be divided into pages and the page fingerprint in L2 will be searched. If the page fingerprint hits in L2, P =P-delta.
4. If all above failed, a search in flash metadata area should be taken.
5. When the space in cache is not sufficient, we have to replace fingerprints in a certain array by comparing the size of L1 with P1 and the size of L2 with P2. If the size of any array is larger than the related P, then it will be cut by replacing fingerprints in this array, and the other one will be enhanced in volume.

This adaptive replacement policy can arrange the size of each array dynamically to get the desired hit ratio according to the nature of DL-Dedupe and the feature of workload. The performance enhanced by adaptive replacement will be demonstrated through experimental result in the following part.

4 Evaluation

In this section, we evaluate our design via a series of comprehensive trace-driven simulations and compare our proposal with the state-of-the-art page deduplication policy. We have implemented DL-Dedupe in a simulator designed by ourselves, and the experiments are conducted on server of HP ProLiant DL388 G7. The workloads downloaded through the trace website [11] have been described in section 2. The most critical parameters that determine the experimental results are redundant rate and

cache hit ratio. The redundant rate is determined by characteristic of workloads and the chunk size, while the hit ratio depends on the cache replacement policy and cache size. Besides the comparison to page deduplication policy, we will also investigate how the chunk size and cache size affect the performance.

4.1 Effectiveness of Deduplication

The performance of DL-Dedupe is effected by hit ratio of both chunk and page fingerprints. As all fingerprints are stored flash, all redundant writes will be deleted after DL-Dedupe. And it will be more effective with high effective chunk deduplication.

As we can see in Fig.3, deduplication is highly workload dependent. Take *homes* as an example, 47% of redundant writes will be removed after the two-page chunk deduplication, and nearly 70% of the whole dataset will be deleted after the dual-level deduplication. However, the effectiveness of chunk deduplication is dropping off with the increasing of chunk size. After chunk deduplication, only 13% duplicate writes will be deleted with four-page chunk size, and the percentages will come down to 8% and 1% with 8-page and 16-page chunks. The result is not so desirable because of the high in page but low in chunk redundancy of the workloads. As we have analyzed, the redundancy of *mail* is higher than the other two, and *web-vm* is lowest. The effectiveness almost follows the law that *mail* is higher than others and *web-vm* is the least effective. To conclude the effectiveness of DL-Dedupe, we can tell that it will be more effective in high redundant workload.

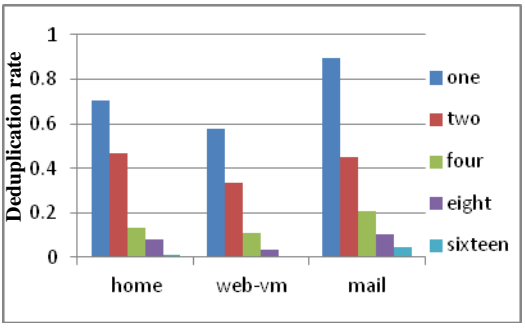
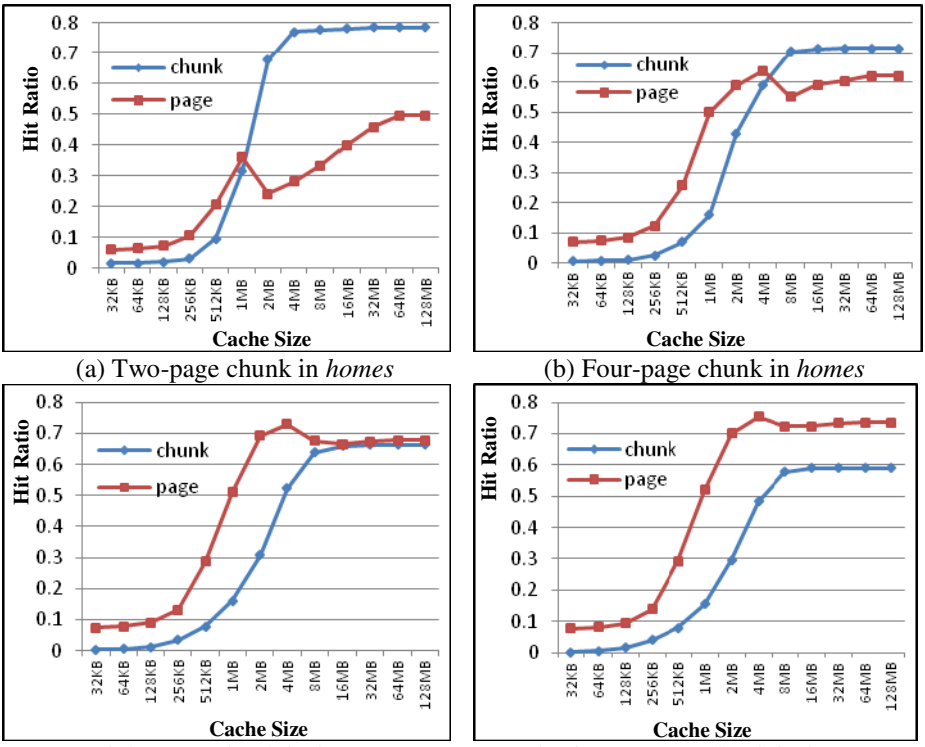


Fig. 3. Deduplication rate of different chunk size (2-16) in three workloads, and one means the page deduplication after chunk deduplication

4.2 Influence of the Cache Size

The adaptive cache replacement policy we have proposed is specially designed for DL-Dedupe to improve cache hit ratio thus improving performance. Though the size of DRAM is determined by manufacturer, we can allocate for a part of it as cache, thus obtaining the optimal performance with minimal cache size. We will evaluate the cache replacement policy in occasion of different cache sizes and different chunk sizes. The hit ratio in cache of both chunk and page are uprising with the increasing of the cache size, until it reaches a certain value that their lines become gentle, so that cache size will impact on the deduplication, especially when the cache size is small.

Chunk hit ratio drops off with the increasing of chunk size while page hit ratio is ascending because of the interaction between chunk and page fingerprints. This interaction is especially obvious in *mail*. However, it is gentle in *web-vm* and *homes*, so that we take *mail* as an example. We can see that in Fig.4 there is a sudden raise in chunk hit ratio with the increasing of cache size, whatever the chunk size is. There will be also a relatively sharp falling in page hit ratio at the same time, because chunk deduplication has deleted much of the redundant data so that page redundancy is low. On the other hand, page hit ratio goes up when chunk hit ratio comes down because it will handle the write operations chunk deduplication left behind. This is the phenomena of interaction between chunk and page deduplication we have noticed.



(a) Two-page chunk in *homes*

(b) Four-page chunk in *homes*

(c) Eight-page chunk in *homes*

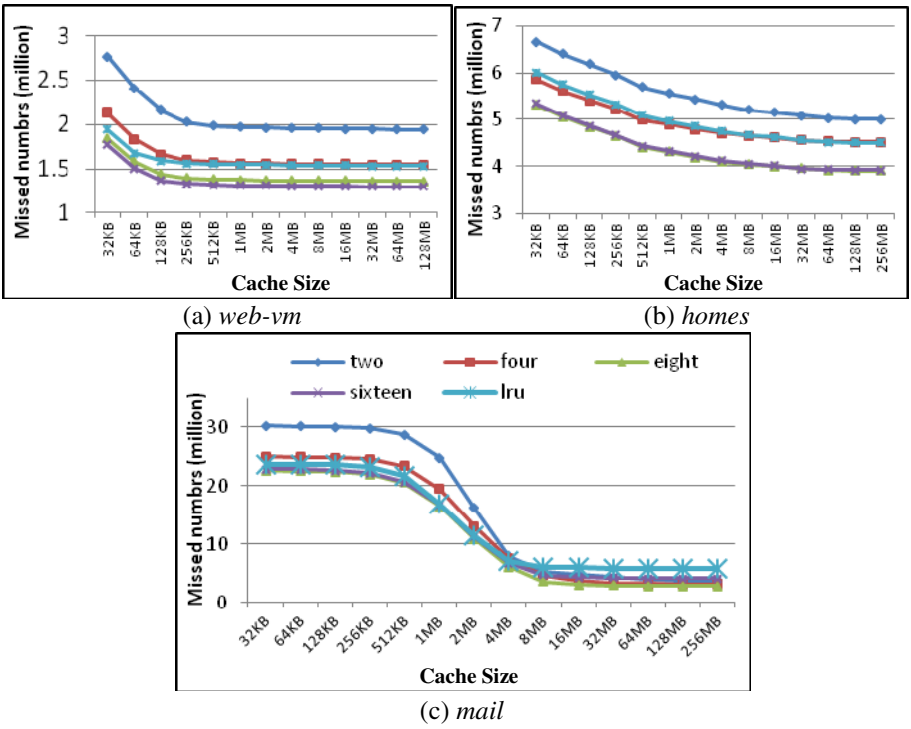
(d) Sixteen-page chunk in *homes*

Fig. 4. The interaction between chunk and page will keep the balance of the performance

4.3 Performance Advantages of DL-Dedupe

In order to evaluate our DL-Dedupe, we compare it with a state-of-the-art page deduplication scheme within SSDs. When it fails to get the fingerprint in cache, an access to flash will be taken which causes the most part of latency. We evaluate the latency of both DL-Dedupe and the page deduplication policy by calculating miss times in cache, which is also equal to access numbers in flash. The results of three workloads are illustrated in Fig.5. As we can see, DL-dedupe gains little for the performance when

cache size is small. It becomes better with the increasing of the cache size, and eight-page chunk of them do the best except in *web-vm*. The redundancy in *web-vm* is not so high that the advantages of DL-Dedupe cannot show in full-scale.



(a) *web-vm* (b) *homes*

(c) *mail*

Fig. 5. SSDs employing our DL-Dedupe miss less than the page deduplication policy in some cases depending on the chunk size and workload redundancy

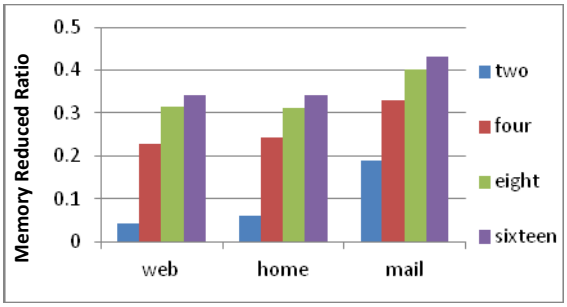


Fig. 6. SSDs employing our DL-Dedupe with different sizes of chunk reduce much memory consumption compared to page deduplication policy

DL-Dedupe is also designed for less metadata and memory overhead. We evaluate the memory consumption by calculating the fingerprints finally stored in the system. Fig.6 shows the fingerprint numbers of different workloads in different chunk sizes to

be stored. The volume of metadata is decreasing with the increment of chunk size, because write requests are divided into bigger chunks, thus the number of fingerprints is reduced. So sixteen-page chunk will reduce the most memory space, thus 34% memory space in *web-vm*, 35% in *homes* and 43% in *mail* are saved. However, there is a trade-off between redundancy and metadata number, so that optimal chunk size will be highly workload dependent. For example, the chunk size in *homes* to gain the best performance is four-page, while it is eight-page in *mail*.

5 Related Work

ChunkStash [14] designed a flash-assisted inline deduplication system using a chunk metadata store on flash. It is the first work to accelerate the search speed of inline deduplication with the help of flash memory. After that, dedupv1 [15] comes out to use the good points of SSD and avoid random writes inside the dada path to improve the throughput compared to disk-based systems. ÄFTL [16] is based on the out-of-place update character to store the delta between new data and old data to reduce data written to flash. However, much redundant data which share the same content without same addresses will be missed.

CAFTL [17] can reduce write traffic to flash memory by removing unnecessary duplicate writes and can also substantially extend available free flash memory space by coalescing redundant data in SSDs. However, the indirect mapping policy is so complicated to complete.CA-SSD [13] is another design on how to utilize deduplication in SSDs, with some extra hardware into SSD to take advantages of value locality. J. Kim [18] do some research on the effect of data deduplication in SSD based on both CAFTL [17] and CA-SSD [13]. Both hardware and software implementation are designed to evaluate the effect of deploying deduplication in SSD. They finally evaluated that it is meaningful.

We propose DL-Dedupe, which is different from all previous work, as it takes sweet spots of not only SSD and deduplication but also the chunk and page deduplication. Besides that, hybrid mapping and adaptive cache replacement are efficient in improving the performance. And the experimental results show that DL-Dedupe can significantly reduce data written to flash and improve the deduplication efficiency and memory availability at the same time, so that lifespan and capacity of SSD can be enhanced significantly.

6 Conclusions

In order to foster strengths and circumvent weaknesses of both SSD and deduplication, we have proposed DL-Dedupe, a dual-level deduplication policy for flash-based SSDs to reduce the volume of data written to flash so that reliability and lifespan of SSD can be enhanced. A specific hybrid mapping policy is designed in DL-Dedupe to minimize the searching overhead. The novel adaptive cache replacement in DL-Dedupe can significantly improve the hit ratio in cache thus reducing the access to flash and optimizing the performance. DL-Dedupe has been evaluated via trace-driven simulation, whose results demonstrate that DL-Dedupe can significantly reduce writes issued to flash and improve the access performance furthermore.

Acknowledgement. We are grateful to the anonymous reviewers for their valuable suggestions to improve this paper. This work is supported by the Key Technology Research of Data Activation under Grant No. 2011AA010502, the National Natural Science Foundation of China under Grant Nos. 61025009, 61232003, 61120106005, 61170288.

References

1. Solid state 101 – an introduction to solid state storage (EB/OI) (April 17, 2009), http://www/snia.org/forums/sssi/SSSI.Wht.Paper.Final.pdf
2. Yinjin, F., Hong, J., Nong, X., Lei, T., Fang, L.: AA-Dedupe: An Application-Aware Source Deduplication Approach for Cloud Backup Services in the Personal Computing Environment. In: CLUSTER 2011 (2011)
3. Richard, V., Carl, O., Matthew, E.: Big Data: What It Is and Why You Should Care. Framingham: IDC (White Paper), 1–14 (2011)
4. Quinlan, S., Dorward, S.: Venti: a new approach to archival storage. In: FAST 2002. USENIX Association (2002)
5. Kulkarni, P., Douglis, F.: Redundancy elimination within large collections of files. In: USENIX2004. USENIX Association (2004)
6. Guo, F., Efstathopoulos, P.: Building a high-performance deduplication system. In: USENIX ATC 2011. USENIX Association (2011)
7. http://www.isuppli.com/Memory-and-Storage/MarketWatch/
8. Lee, S., Park, D., Chung, T., Lee, D., Park, S., Song, H.: FAST: An FTL Scheme with Fully Associative Sector Translations. In: UKC (August 2005)
9. Lee, S., Shin, D.: LAST: locality-aware sector translation for NAND flash memory-based storage systems. SIGOPS 42(6) (2008)
10. Gupta, A., Kim, Y., Urgaonkar, B.: DFTL: A Flash Translation Layer Employing Demand-based Selective Caching of Page-level Address Mappings. In: ASPLOS 2009 (2009)
11. SNIA. IOTTA repository (January 2009), http://iotta.snia.org/
12. Ricardo, K., Raju, R.: I/O Deduplication: Utilizing Content Similarity to Improve I/O Performance. In: FAST 2010. USENIX Association (2010)
13. Gupta, A., Pisolkar, R., Urgaonkar, B.: Leveraging Value Locality in Optimizing NAND Flash-Based SSDs. In: FAST 2011. USENIX Association (2011)
14. Debnath, B., Sengupta, S.: ChunkStash: Speeding up Inline Storage Deduplication using Flash Memory. In: USENIX ATC 2010 (2010)
15. Meister, D.: dedupv1: Improving Deduplication Throughput using Solid State Drives (SSD). In: MSST 2010 (2010)
16. Wu, G., He, X.: ΔFTL: Improving SSD Lifetime via Exploiting Content Locality. In: EuroSys 2012, Bern, Switzerland, April 10-13 (2012)
17. Chen, F., Luo, T., Zhang, X.: CAFTL: A content-aware flash translation layer enhancing the lifespan of flash memory based solid state drives. In: FAST 2011, San Jose, CA, February 15-17 (2011)
18. Kim, J., Lee, C., Lee, S.: Deduplication in SSDs: Model and Quantitative Analysis. In: MSST 2012 (2013)

SFCM: A SSD-Friendly Cache Management Policy for Hybrid Storage Systems

Jiangtao Wang, Wenyu Lai, and Xiaofeng Meng

Renmin University of China, Beijing, China
{jiangtaow,xiaolai913,xfmeng}@ruc.edu.cn

Abstract. With the growing popularity of flash memory, solid state drivers (SSD) based on flash memory have been widely used in various kinds of applications. Compared with conventional magnetic disk, SSD can provide higher access bandwidth and lower access latency. However, it will not completely replace the disk as the secondary storage in the short run due to its inherent properties such as asymmetric read/write, high price of per gigabyte etc. Integrating SSD and magnetic disk together can make full use of different performance advantages, so as to obtain good high performance and low cost. This paper proposes SSD-friendly cache management scheme (SFCM) that use SSD as a cache layer between main memory and magnetic disk. For the pages evicted by buffer manager, SFCM conditionally caches them in SSD according to the state of the page and the different replacement cost. Due to the quick access performance of SSD, SFCM can improve the performance of cache management when the destination page resides in SSD. In view of the poor random write performance for SSD, SFCM adopts flash-aware algorithms and data structures to manage the data stored in SSD, which takes care of write patterns and request types' impacts on system performance. Furthermore, the study proposes a flash-friendly buffer replacement policy which considers the asymmetric I/O cost and the workload access features. We implement the scheme on the multi-level cache storage system based on a simulation platform and evaluate the performance. The experimental results show that SFCM can significantly reduce system response time and disk I/O cost. Adding a flash-based SSD as cache extension can make up the performance gap between memory and disk.

Keywords: Solid State Driver, Cache, Flash-Aware, Replacement Cost.

1 Introduction

With the high-up development of flash memory techniques, flash-based solid state drive (SSD) has been extensively used in personal computing and the server market. Compared with magnetic hard disks, SSD enjoys faster access speed, higher shock resistance, and lower power. It is making significant inroad into enterprise applications with continuously growing capacity and dropping price [1, 2]. In some cases, SSDs have been used as the main storage device for data management [3]. However, using SSDs to entirely replace hard disks is not cost-effective. One reason is that

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 16–27, 2013.

SSDs are more expensive than hard disks when measured the price per gigabyte capacity[7], another important reason is that the current relatively small capacity of SSDs is hard to meet the demand for storage massive data storage. Therefore, integrating SSDs and hard disks together may be an economical way to deal with large-scale data-intensive applications.

In this paper, we propose a novel scheme called SFCM, where SSD is used as an extension cache layer between main memory and hard disk. The design goal of SFCM is to improve the performance of cache management when the destination page missed in main memory but resided in SSD due to its quick access performance. SFCM has a 3-tier storage hierarchy, which is composed of main memory, SSD, and hard disk. When the accumulated data pages fill up the memory buffer, SFCM adopts operation-aware replacement policy to select a victim page and conditionally caches it to the SSD. Our replacement algorithm designed for memory buffer can efficiently decrease the cost of SSD I/O. For SSD extension cache, due to its special characteristics (such as erase-before-write, out-of-place update), frequent random writes may lead to more erase operations and seriously aggravate the storage performance of SSD[9]. Data management policy proposed in this article optimizes write pattern and tries to reduce random write operations. When an evicted page is identified to reside in SSD, SFCM locate a fixed-size storage segment to hold it, which ensures write operation do not occur a scattered random write. When the available space in SSD is insufficient, SFCM computes the replacement priority for all segments according to their access frequency and the number of valid pages, and finds a victim segment with lowest priority. The page which is invalid or rarely accessed will be evicted out of the SSD. If the page evicted from SSD is dirty, SFCM will write it back hard disk. We summarize our contributions as follows:

- We present a flash-based multi-level cache scheme, which employs SSD as an extension of DRAM to handle frequent access requests.
- We describe a cost-aware replacement policy for memory buffer pool. Considering different page replacement costs and workloads access pattern, SFCM splits the buffer into two LRU queues, and selectively swaps a page out of the main buffer according to the length ratio of two LRU queues.
- We propose a flash-friendly data management policy for SSD. SFCM organizes the SSD cache into a segment queue, each write operation is confined within a segment, and do not occur scattered random write.
- We demonstrate the effectiveness of SFCM by comparing with a state-of-the-art cache extension approaches presented in a recent study. The evaluation results show that SFCM can significantly shorten the running time.

The rest of this paper is organized as follows: Section 2 describes prior work related to cache extension approaches for flash-based heterogeneous storage system. Some critical issues of SFCM, including system overview, buffer management and SSD data management, are introduced in Section 3. Section 4 gives the results of our experiment evaluation. Finally, Section 5 concludes this paper.

2 Related Work

As a novel storage medium, flash-based SSD is being paid more and more attention in recent years, lots of work emerged to overcome the I/O bottleneck of magnetic disk. FlashStore[4] provides a high throughput persistent key-value store which uses SSD as a cache between DRAM and magnetic disk. It manages key-value pairs in a log-structure on SSD to exploit faster sequential write performance. Using flash-based SSDs as multi-level cache has been reported in [5, 6, 7]. In [5], the authors discuss how a flash memory can efficiently act as an extension cache between the main memory and the hard disk. Then, three extension SSD buffer pool schemes which include inclusive approach, exclusive approach and lazy approach are proposed. In the exclusive scheme, when a page is read from the SSD cache, another copy of the same page will be dequeued from the SSD cache. Under inclusive method, the set of pages cached in main memory is always a subset of the pages cached in SSD. The lazy design, on the other hand, writes the pages evicted from memory buffer to the SSD first, and then lazily copies pages from the SSD to the disk (if the page is dirty).

Canim et al. [6] proposed a way of using an SSD as a second-level cache, which is called Temperature-Aware Caching (TAC). TAC monitors the accesses to pages at the granularity of a block (a block includes 32 contiguous pages) and identifies hot data blocks. When a disk read operation occurs, TAC computes the temperature of block in which the destination page resides, if the block is identified to a hot region, it will be cached in SSD. When a dirty page is evicted from the main memory, TAC writes it to both hard disk and SSD simultaneously. TAC adopts a write-through policy to deal with dirty pages, and do not reduce the total number of write operations to disk. Using flash memory as extension cache is also discussed in [7], which proposes two novel algorithms for flash cache management in order to improve the transaction throughput and shorten the recovery time. The authors adopt FIFO replacement policy to manage the data pages cached in SSD, when the SSD receives a write request, it will be done sequentially through an append-only fashion. In addition, the authors take advantage of the non-volatility of flash memory to accelerate database recovery by adding checkpoint mechanism for the metadata which records the mapping information between disk and SSD. In [7], the high cost of maintaining cache metadata and reclaiming invalid space in flash memory may degrade the overall performance.

3 The Implementation of SFCM

3.1 System Overview

SSD serves as a cache layer between DRAM and hard disk in our SFCM scheme. Figure 1 describes the main components and data interactions between heterogeneous storage mediums. In this figure, arrows represent the flow of data pages. The main memory buffer is consulted when the buffer manager gets a request for a page. If the data do not reside in memory, its corresponding page will be fetched from SSD or disk. When the main memory has no available space, SFCM selects a victim page to

swap out. Although the size of SSD caches is always much larger than main memory, however, along with the accumulation of the pages evicted from main memory, the SSD will be filled, so that the storage space of the SSD need to be organized carefully for better utilization. For this reason, data replacement policy on SSD proposed in SFCM will evict some victim pages to discard or write back disk. When a dirty page is evicted from the buffer pool, it will be written to the SSD first, and lazily write back to disk. In the next section, we describe the implementation details of the SFCM architecture. We first introduce buffer management policy in memory, and then present the replacement policy and admission policy on SSD. The SSD management policy is used to determine when and which data pages should be staged in or out of the SSD cache.

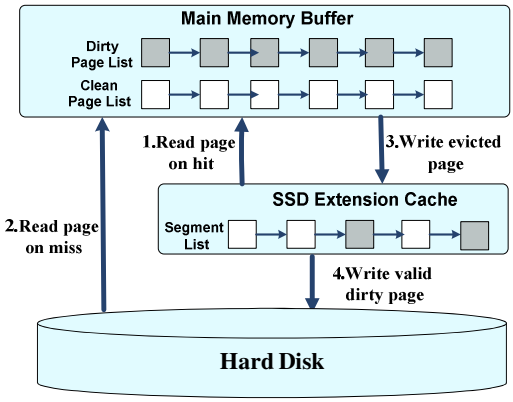


Fig. 1. System overview

3.2 Memory Management

An effective buffer management policy can reduce the buffer miss rate and enhance the overall performance of the database system. However, maintaining a high hit ratio may result in a sub-optimal I/O performance in flash-based systems. This is because that write operation of flash memory is an order of magnitude slower than read operation on the flash memory, and the performance benefit of reserving dirty pages tends to be higher than that of reserving clean pages. Considering the asymmetric read and write operation costs of flash disks in terms of performance, we split the buffer into two LRU queues, say Q_C and Q_D. Q_C is used to manage the clean pages, Q_D is responsible for the dirty pages. The size ratio of Q_C and Q_D is given by Formula 1:

$$\frac{length(Q_c)}{length(Q_d)} = \frac{C_r}{C_r + C_w} * \lambda = \omega \tag{1}$$

Where C_r and C_w stand for the time cost of a read and a write operation respectively. λ is a tuning parameter which considers the impact of access pattern. If the workload is read-intensive, the value of λ will be augmented in order to keep more

memory space for Q_C. When the workload is update-intensive, we reduce the value of λ to add available memory space for Q_D. The detailed buffer management algorithm of SFCM is listed as Algorithm 1.

Algorithm 1. **SFCM _bufferM(page *p*,buffer *B*)**

```
if (p resides in B)
  find the requested page p ;
else
 { if (buffer is full)
    {
     VictimPage=M_SelectVictimPage();
      flush VictimPage to SSD;
    }
   if( p resides in SSD)
        load p from SSD;
   else
       load p from Disk;
 M_SelectVictimPage()
 {  Lc=Length (Qc) ;
    Ld=Length (Qd) ;
    if (Lc/ Ld> ω )
       return the tail of Qd queue;
    else
       return the tail of Qc queue; }
```

When the buffer manager receives a write request, it searches the buffer B first, if the page is found (cache hit), SFCM adjusts the LRU order of p in Q_c or Q_d and returns p. Otherwise, the buffer manager allocates a new slot for page p and loads the requested page from SSD or disk (lines 4-17). A page replacement is needed when the buffer is full and cannot accommodate a new page data (lines11-17). Let L_c and L_d be the current queue length of Q_c and Q_d , respectively. To find a proper victim page, SFCM calculates the replacement condition which is used to decide which queue will be selected to provide the victim candidate, and chooses the LRU page in the corresponding queue. SFCM proposes a flash-aware replacement policy, it gives dirty pages more opportunities to stay in the buffer, which contributes to reduce write operations in flash memory and improve the overall performance.

3.3 SSD Data Management

Using a flash SSD as an extended cache is an efficient and cost-effective way to improve the performance of data management system. However, flash memory is not perfect due to its inherent characteristics, some operations which include random write and small granularity write are not efficient for flash-based SSD. We must try to avoid or reduce the negative effects caused by these operations.

1) Design Principles: The principal problem that should be carefully considered in flash-based caching system is how and which data pages should be written or swapped out of the SSD cache. In this section, several issues proposed to solve the problem mentioned above are summarized as follows:

- When the requested page is not found in memory buffer, we need to search the page in the SSD cache. We usually maintain a translation table to support the search operations. However, the size of the SSD cache is much larger than the memory buffer, so we must design and realize a light-weighted index structure to quickly locate the destination page.
- For most flash memory, random write is slower than the other access pattern approximately by an order of magnitude. Some studies [8, 9] show that a scattered random write which writes a page in a large storage space performs badly. Small random writes are helpful to eliminate I/O bottleneck caused by inefficient write pattern. We design a novel storage strategy for SSD cache which confines the write operation to a relative small disk area to avoid scattered random write.
- As for a SSD cache extension, if the incoming page evicted from main memory is dirty and its old copy exists in the flash cache, we need to rewrite the latest version. Due to the erase-before-write limitation of flash memory, flash memory implements out-of-place update policy to overwrite a page, which incurs a lot of random write and hinders the improvement of SSD-based multi-level cache system. It is necessary to decrease the expensive cost of the overwrite operation. We adopt a flash-friendly write pattern which invalidates the old page and sequentially appends the new page in a fixed size storage segment to reduce the number of write and erase operations on SSD.

2) Data Structure: Cache management policy proposed in our work uses the following five data structures.

SSD Buffer Queue: We split all the SSD buffer pool into multiple fixed-size data segment, each segment is composed of 256 data pages. All the segments are maintained implicitly as a circular queue.

Translation Table: This translation table is a metadata file which records address mapping information for the pages stored in the SSD buffer. Each table entry includes segment_ID, dirty flag that indicates the state of the evicted page, validation flag that describes the page whether is valid or not, access frequency, some pointers such as next segment pointer, and etc.

Hash Table Index: The index structure maintained in memory, for the entries in the translation table, is used to support a fast lookup on the SSD buffer. Each index entry has a pointer which points to a mini hash table with the goal of fast locating one flash page.

Victim Identifier Heap: When SSD usage exceeds a certain threshold, the SSD buffer pool has not enough free frames to hold the page evicted from main memory, we must select some victim pages to stage out of SSD cache and recycle previous used pages. To quickly identify these victim pages, a victim identifier heap is used. The heap array records the replacement weight of each segment, and the heap root has the lowest priority. With help of the heap array, replacement policy proposed in our work can quickly determine which pages will be evicted from SSD.

Free Space Bitmap: A free space bitmap is used to indicate all the available segments on SSD after freeing invalid data and replacing cold pages.

3) Data Access on SSD: Once memory access failure occurs, SFCM need to visit SSD cache and search the requested page through the translation table. When the memory buffer manager uses replacement policy to evict a victim page, SFCM need to adopt admission policy to selectively cache pages in the SSD. The detailed procedure is shown in Algorithm 2.

Algorithm 2. accessSSD(page *p*,operation *t*)

```
if (t is fetch)
  {   s=locate_segment(p);
      p.frenquency++;
      s.frenquency++;
      return p;}
else
   {     s=locate_segment(p);
         if(p is dirty)
           {   p.frenquency+= int(Cw/Cr);
               s.frenquency+= int(Cw/Cr);}
             /*Cw,Cr is the time of a write/read operation on SSD*/
         else
         {   p.frenquency++;
               s.frenquency++;}
         if(exist an old copy of p)
             invalidate the old page;
         if(SSD is full)
                evictVictimPages();
               write page p to SSD;
    }
```

As for a read I/O request, the algorithm will update the access frequency of the destination page and segment in which the page resides, respectively (line2-4). If a write I/O request occurs, in order to augment the replacement weight of the destination page, its access frequency is increased by m, where m is equal to the ratio of C_w and C_r (line9-10). The reason is that a dirty page needs to be written back hard disk which will incur an expensive disk I/O, and should have priority over clean one with respect to caching in SSD buffer pool. Overwriting an earlier version data page issues a random write for an (logically) in-place replacement. In order to reduce SSD I/O cost, the new data page will be appended to the same segment, its previous copy is invalidated, and be recycled later through replacement policy (line15-19).

4) Replacement Policy: The extended SSD cache is populated with pages evicted from the memory. All identified incoming pages will be admitted, so long as the SSD cache has free frames. Subsequently, a replacement policy is used to evict some cold pages on SSD. Much replacement algorithms have been conducted on buffer cache management, such as LRU which has been used widely for buffer cache management. These algorithms were designed for single-level buffer caches which access exhibit a high degree of temporal locality. In SFCM, memory and SSD form a multi-level

buffer cache hierarchy, where SSD serves as second-level buffer cache. Previous studies show that second-level buffer cache has weak locality, and locality-based replacement algorithms are not suitable for second level buffer cache [10, 11]. For low level cache, the strong locality has been filtered by the high level buffer caches. Zhou et al. [5] observed that pages that are accessed more frequently contribute more data. Therefore, replacement algorithm needs to give different priorities for pages with different frequencies. Some existing cache extension approaches adopt various variant of LRU algorithm to manage data, these algorithms trigger many random writes that require SSD pays for expensive cost. Considering access frequency and pattern, we propose a SSD-friendly replacement policy (described in Algorithm 3). When there is not enough available space in the SSD buffer, replacement process is carried out. First, SFCM updates the replacement priorities of all segments according to the formula 2, and finds a victim segment with the lowest priority (lines 1-8):

$$priority(s) = \frac{frequency(s)}{invalid\_count(s)} \tag{2}$$

In Formula 2, *frenquency(s)* stands for the total access frequency of segment *s*, *invalid_count(s)* maintains the number of invalid pages in segment *s*. A segment with low access frequency and more invalid data page will be chosen for victim segment. In order to reduce cumulative effect and prevent rarely accessed pages from staying in flash cache for a long time, we also consider the aging mechanism, the replacement priority of each segment, which is not accessed recently, would be degraded to a lower priority (Line 4-5). Next, pages having the lowest frequency in current victim segment are removed from SSD until the frequency of candidate pages exceeds $T_{frequency}$ which indicates an access frequency threshold (line 7-14). If the victim page is dirty, its copy is propagated to disk. All the invalid pages in the segment are dequeued from the SSD cache in a batch.

Algorithm 3. evictVictimPages()

```
Snum=total number of segment;
for (i=0; i<Snum;i++)
  {
   seg_weight[i]=seg_frenquency(i)/seg_valid_count(i);
   if (seg_periodfre(i)>a predefined threshold)
     seg_weight[i]= seg_weight[i]* φ;
}
Segment S=select VictimSegment with the smallest weight ;
for(j=0;j<S.pagenumber; j++)
  { if(page[j].invalid==1)
      discard page[j];
    else
     { if(page[j].frequency<T_frequency)
            if (page[j] is dirty)
               write back disk;
          discard page[j]}
```

4 Performance Evaluation

In this section, we describe the experiment environment and performance results. Section 4.1 presents the experiment setup and how the experiments were performed. The experiment results and some analysis are shown in Section 4.2.

4.1 Experimental Setup

In order to verify the effectiveness of SFCM compared to previously proposed methods, we design and implement a hybrid storage simulation system based on DiskSim simulator [12] and its SSD extension patch [13]. It provides functions that count the number of physical write and erase operations on flash, which is helpful to evaluate the performance of flash-friendly replacement policy proposed in SFCM. We collected two types of traces by running the TPC-C benchmark [14] on PostgreSQL DBMS. The statistics of the two traces are shown in Table I, where x%/y% in column "Read/Write Ratio" means that for a certain trace, x% of total requests are about read operations and y% about write operations.

Table 1. The statistics of the traces used in our experiment

Type	Total Requests	Read	Write	Read/Write Ratio
T1	2193476	1805230	388246	82.3%/17.7%
T2	1988563	1262737	725826	63.5%/36.5%

The simulator is based on a SLC flash, the operation latency is calculated by applying a weight to each operation, all the configurations are summarized as follows: the weight for read, write and erase operations are 25, 200 and 1500, respectively. The page size is set to 2KB, and the block size is 128 KB.

4.2 Performance Results and Analysis

We choose the following metrics to evaluate the performance of SFCM : (1) running time on different traces;(2) number of physical write operations; (3) SSD hit ratio with different cache size.

1)Running Time: We replay two trace files on either a flash memory SSD or a hard disk, which can facilitate us understand the performance impact by flash extension cache. As shown in Figure3, using SSD as secondary storage can significantly shorten the running time. We run two trace files with the flash cache size varying from 5 to 30 times the size of the main memory, and analyze the performance impact caused by SFCM with respect to running time. To demonstrate its electiveness, we also implement a state-of-the-art cache extension approaches presented in a recent study (referred to as Lazy_S) which using LRU as replacement policy [5]. Fig.5 and 6 show the comparison results between Lazy_S and SFCM.

It will be observed from these figures that our method performs better than Lazy_S for all flash cache sizes, which mainly owe to the decline of random write operations

on SSD. Another important reason is that Lazy_S fairly process the clean and dirty pages cached in SSD, which incurs expensive disk I/O cost when write dirty pages back disk. Specially, for a flash cache size more than 15 times that of the memory, the running time of Lazy_s method remains flat, and has no any significant improvements. This is because Lazy_S has make the most strenuous effort to process frequently random write as more data pages were stored in SSD. In this case, flash cache becomes a performance bottleneck. In contrast, under the SFCM method, even if the trace is T2 which has more write operations than T1, the performance with respect to running time continues to improve with the increase of the flash memory cache size.

2)Physical Write Operation: We run the experiment on two trace files, and count the number of physical write operations on SSD. For different flash and RAM sizes, the number of write operations has been normalized with respect to the Lazy_S method. Similar results were measured for our experiment, due to space limitations we only report the results for trace T1(shown in Fig.4). It can be seen from the figure that SFCM consumes less write operations than Lazy_S, one reason is that SFCM try to avoid the random write within a large area, another is that we adopt cost-aware buffer replacement policy, these flash-aware write patterns decrease the number of write/erase operation when doing space reclamation and garbage collection.

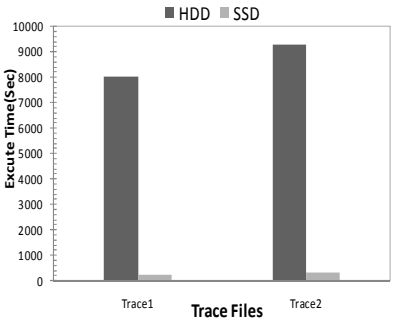


Fig. 2. Run time in a single device

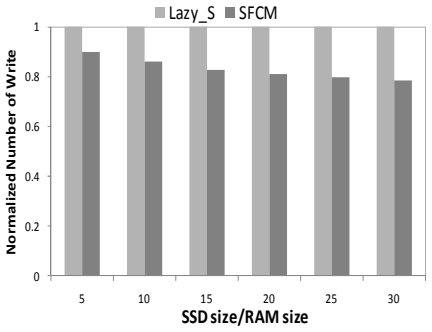


Fig. 3. The number of SSD write on T1

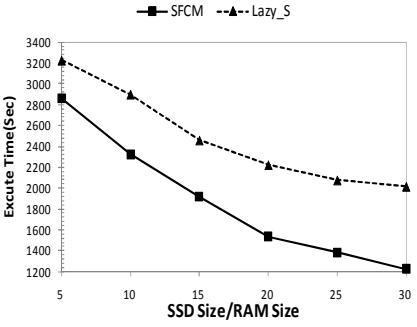


Fig. 4. Run time on T1

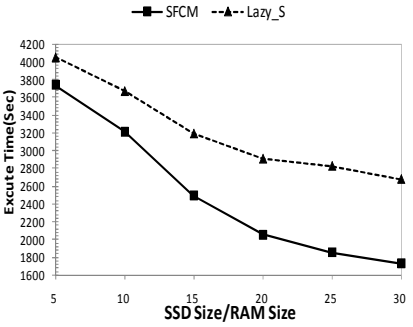


Fig. 5. Run time on T2

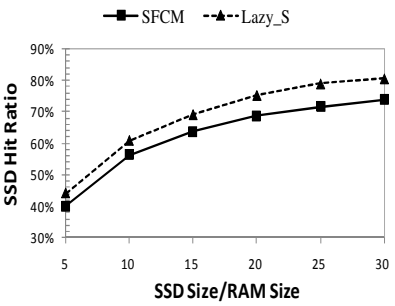


Fig. 6. SSD hit ratio on T1

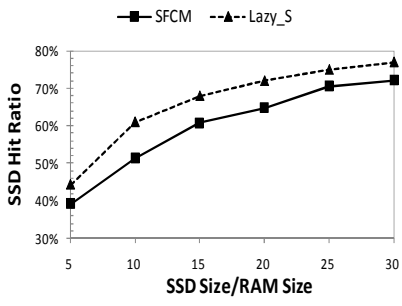


Fig. 7. SSD hit ratio on T2

3)SSD Hit Ratio: Fig.7 and 8 show the result of cache hit ratio on SSD for trace T1 and T2. We can see that the hit rate grows with the increasing size of the SSD cache. Compared with the Lazy_S method, the hit ratios of SFCM are slightly lower than those of Lazy_S. This is because, when the SSD buffer pool is managed by SFCM, the SSD cache may hold some invalid data pages before evicting victim pages. Thus, Lazy_S can utilize the space of the flash cache more effectively than SFCM and maintain a higher hit ratio. It is worth noting that high hit ratio on SSD sometimes does not necessarily bring high system performance, as we see in Fig.5 and 6, the high hit ratio for Lazy_S does not always result in a lower I/O cost.

5 Conclusion

In this paper, we propose a SSD-friendly multi-level caching policy, referred to as SFCM, in such a way that a flash-based SSD is used as an extension buffer pool to cache data pages evicted from memory buffer. By means of cost-aware memory buffer management algorithm, SFCM reduces the number of physical write operations and enhance the endurance of SSDs. Depending upon flash-friendly replacement policy designed for the SSD cache, SFCM maintains a relatively high hit ratio and shortens execution time. We conduct a performance evaluation of our method based on two different trace. The results show that the running time of disk-based storage system is approximate 8 times longer than SFCM for a flash cache size more than 30 times that of the memory cache. And the running time of SFCM is only 5 times longer than based-SSD storage system. The results demonstrate that SFCM is a cost-effective method to achieve a better overall performance.

Acknowledgements. The research was partially supported by the grants from the Natural Science Foundation of China (No,60833005,61070055,91024032, 91124001);the Research Funds of Renmin University of China(No:11XNL010);the National High Technology Research and Development Program of China(No.2012AA010701).

References

1. Canim, M., Bhattacharjee, B., Mihaila, G.A., Lang, C.A., Ross, K.A.: An Object Placement Advisor for DB2 Using Solid State Storage. Proceedings of the Very Large Data Base (VLDB) Endowment 2(2), 1318–1329 (2009)
2. D. White Paper: MySpace Uses Fusion Powered I/O to Drive Greener and Better Data Centers.(EB/OL), http://www.fusionio.com/case-studies/myspace-case-study.pdf
3. Lee, S., Moon, B., Park, C., Hwang, J., Kim, K.: Accelerating In-Page Logging with Non-Volatile Memory. Data Engineering 33(4), 41–47 (2010)
4. Debnath, B., Sengupta, S., Li, J.: Flashstore: high throught persistent key-value store. Proceedings of the Very Large Data Base (VLDB) Endowment 3(2), 1414–1425 (2010)
5. Koltsidas, I., Viglas, S.D.: Designing a Flash-Aware Two-Level Cache. In: Eder, J., Bielikova, M., Tjoa, A.M. (eds.) ADBIS 2011. LNCS, vol. 6909, pp. 153–169. Springer, Heidelberg (2011)
6. Mustafa, C., George, M., Bishwaranjan, B., Kenneth, R., Christian, L.: SSD Bufferpool Extensions for Database Systems. In: VLDB, pp. 1435–1446 (2010)
7. Kang, W.H., Lee, S.W., Moon, B.: Flash-based Extended Cache for Higher Throughput and Faster Recovery. Proceedings of the Very Large Data Base (VLDB) Endowment 5(11), 1615–1626 (2012)
8. Chen, F., Koufaty, D.A., Zhang, X.: Understanding intrinsic characteristics and system implications of flash memory based solid state drives. In: SIGMETRICS, pp. 181–192 (2009)
9. Bouganim, L., Jónsson, B.T., Bonnet, P.: uFLIP:Understanding flash io patterns. In: CIDR (2009)
10. Muntz, D., Honeyman, P.: Multi-Level Caching in Distributed File Systems-or-Your Cache Ain't Nuthin' but Trash. In: USENIX, pp. 305–314 (1992)
11. Willick, D.L., Eager, D.L., Bunt, R.B.: Disk Cache Replacement Policies for Network Fileservers. In: ICDCS, pp. 2–11 (1993)
12. Ganger, G., Worthington, B., et al.: The DiskSim simulation environment(v4.o) (E13/oL] (September 2009), http://www.pdl.ainu.ed/DiskSim
13. Agrawal, N., Prabhakaran, V., Wobber, T., et al.: Design tradeoffs for SSD performance. In: USENLX, pp. 57–70 (2008)
14. TPC Benchmark C: Standard Specification, http://www.tpc.org/tpcc/spec/tpcccurrent.pdf

HB-Storage: Optimizing SSDs with a HDD Write Buffer

Puyuan Yang, Peiquan Jin, Shouhong Wan, and Lihua Yue

University of Science and Technology of China, Hefei, China
yangpuy@mail.ustc.edu.cn

Abstract. In recent years, flash memory based storage device SSDs (solid state drives) have been regarded as the storage devices of next generation to replace HDDs (hard disk drivers). However, the high price of SSDs, especially those with high performance, results in the situation that SSDs and HDDS are both popularly used in real applications. In order to integrate the merits of SSDs and HDDS, it has become a hot research topic that using HDDs for SSDs to construct a hybrid storage system. The goal of this paper is to use the cheap low-end SSD and HDD to build a hybrid storage system with high efficiency, which is called HB-Storage. HB-Storage considers the characters of SSDs and HDDs, and builds a HDD write buffer to optimize the SSD write request. The write buffer is designed based on the data access load statistics. As a consequence, HB-Storage can utilize the higher read performance of SSDs, and can also improve the random write latency of SSDs. The experimental results show that HB-Storage can maintain a high read performance and significantly reduce the write requests on the SSD, and thus has higher overall performance.

1 Introduction

Flash memory, as a new type of storage media, has received much attention in recent years. Flash memory as well as flash-based solid drives (SSDs) has many advantages compared with magnetic disks, such as high I/O performance, low energy consumption, and so on. However, although flash memory has a lower reading latency compared with magnetic disks, it usually has a higher latency in random writes. From this point of view, we often say that flash memory has asymmetrical I/O performance. In addition, due to the erase-before-write characteristics, flash memory cannot be updated in place, i.e., a write operation would trigger a costly erase operation on flash chips. As the erase count for flash chips is very limited, the random writes are much critical to the reliability of SSDs. In other words, write operations on flash memory will not only influence its performance but also has a big impact on the life of SSDs.

Nowadays, more and more data are produced in the "big data era". As the performance of big data processing is always a key issue in various applications, it has been very common to use high-end SSDs in big data applications. For example, Baidu, a dominant search engine in China, has equipped SSDs in over 1,000 servers to improve the searching performance. High-end SSDs use a variety of hardware optimization techniques [1] to ensure their satisfied I/O performance. However, high-end SSDs are usually very expensive. In particular, the use of high-end SSDs will results in extremely high costs. Therefore, some low-end SSDs are also used in current storage systems. Despite the poor I/O performance of low-end SSDs, the low price is able to promote the popularity of low-end SSDs.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 28–39, 2013.

On the other hand, HDDs have a symmetric I/O performance, there is no limitation similar to the flash write operations, and their price is also lower than that for SSDs. Therefore, building an efficient hybrid storage system, based on an inexpensive low-end SSDs and HDDs, is valuable to users. This scheme can achieve the goal to obtain a larger performance improvement but with a lower monetary cost.

In this paper, we present a hybrid storage system involving an SSD and an HDD write buffer, which is called HB-Storage. HB-Storage is based on the tendency calculation model RWTend, and aims at building a HDD buffer to optimize the write performance for the SSD. The main characters of HB-Storage can be described as follows:

(1) We propose a analysis model for pages' hotness state, called HotnessCalculate. HB-Storage sets a time threshold *t*. A page is regarded as hot when the time interval between its two successive physical accesses is smaller than *t*, otherwise the page is regarded as cold. Only hot pages can be cached on HDD.

(2) We introduce a model to calculate the read and write tendency of pages, called RWTend. The read and write cost of pages on SSD and HDD is calculated by combining the access statistics and performance parameters of SSD and HDD. The write cost would take up major proportion of SSD access cost due to low-end SSD's asymmetry performance. The page access tendency can be clear by comparing the access cost of HDD and SSD, and then HB-Storage executes write and read separately according to the tendency, as shown in Figure 1.

(3) We present an algorithm to maintain the HDD buffer. As shown in Figure 1, it is the write cache operation that a page, with hot access and write tendency, is written on HDD when it is evicted from DRAM, while it is a migration operation that the page is evicted to SSD from HDD. In order to minimize unnecessary migration operation, HB-Storage system, considering the effect of the cold pages is too small, would not move the page from HDD to SSD if the page becomes cold with access tendency unchanged, and would not cache the cold pages on HDD even the pages are read-tendency.

The rest of this paper is organized as follows. Section 2 discusses the realted work. Section 3 will discuss the HB-Storage policy, including the RWTend model and

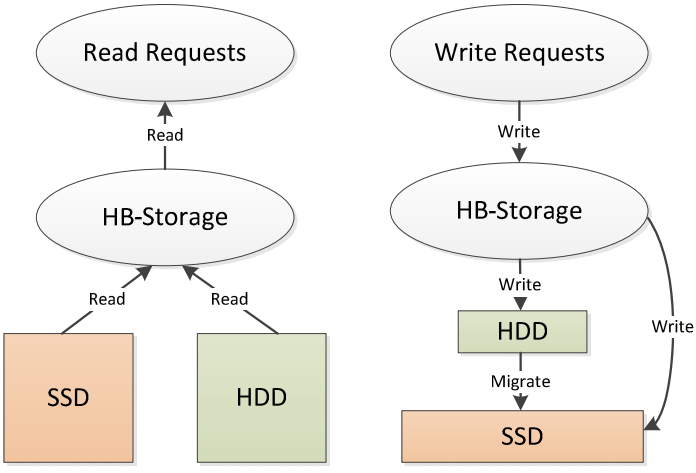


Fig. 1. Read and write operations processing

HotnessCalculate model. Section 4 will discuss the implementation of HB-Storage and experiment. Finally section 5 will summarize the work of this paper.

2 Related Work

Page classification algorithm and migration strategy is a very important part of hybrid storage system [2, 3], Page classification algorithm places the pages by the analysis of access load: the read-tendency pages would be on SSD while the write-tendency pages would be on HDD [4]. In hybrid storage system, the page migrations caused by the change of access load, between the two kinds of storage media, would result in an adverse impact on system performance with additional write cost. It is only when the income is greater than the cost of migration operation that the migration policy is effective. Based on I/O statistics, In [5] the authors proposed a hybrid storage system which calculated the access tendency of page on SSD and HDD according to the page I/O statistics, and then placed the page by the tendency, besides, the migration operation is triggered when the page's access tendency is change. This system is insensitive due to the calculation is based on long-time statistics. This paper will use a similar method to calculate the page access tendency for finding the write-tendency pages to buffer on HDD.

On the other hand, because of the I/O performance of flash memory asymmetry, the concept of access hotness (hot/cold) is generally used by the flash-related researchers [6-9]. In general, the page is regarded as hot when accessed frequently, and is considered as cold when non-frequently accessed. Related research works takes the policy of buffering the hot pages on SSD to reduce the physical write operations to SSD. Some researchers have tried to apply the policy to a hybrid storage area and get some meaningful research results [10, 11]. This paper uses hotness sensitive method to identify the frequently updated page to cache on HDD.

HDD can be taken as SSD's buffer for write requests because of HDD write performance advantage, while read requests to SSD need no buffer because of SSD read performance advantage. Therefore, HB-Storage has asymmetric read and write processing mechanism which separates the read and write request processing flows. The write-tendency pages would be cached on HDD after updated while the read-tendency pages are not need to cache on HDD.

3 HB-Storage

3.1 Architecture of HB-Storage

HB-Storage system architecture is shown in Figure 2. In Figure 2, the arrows indicate the flow of the data page, and DRAM is managed by LRU policy. When a page is accessed, its information about hotness state and access time is firstly updated, which is discussed in section 2.2. When a page is evicted from DRAM, its access tendency is updated by the model RWTend which is discussed in section 2.3. The pages with hot state and write-tendency would be written buffer to HDD, while the pages without satisfied condition would be directly written to SSD. Besides, if the model RWTend identifies that a page of HDD becomes read-tendency, the page is migrated to SSD, which expresses the page is 'evicted' from HDD.

To mark the storage location of the page, HB-Storage builds page address index for SSD and HDD. When the access requests arrive, HB-Storage firstly searches the page in HDD-index, if the target page is not there, then searches it in SSD-index. When a page of SSD is cached on HDD, its flag stays in SSD-index and is inserted to HDD-index. On the other hand, when a page of HDD is 'evicted' to SSD, its flag is removed from HDD-index and its old flag in SSD-index is updated, which realize the migration. In short, when the page is cached in HDD, its information is stored in both SSD-index and HDD-index, and the data version of the page on both devices is not synchronous with only data on HDD being valid.

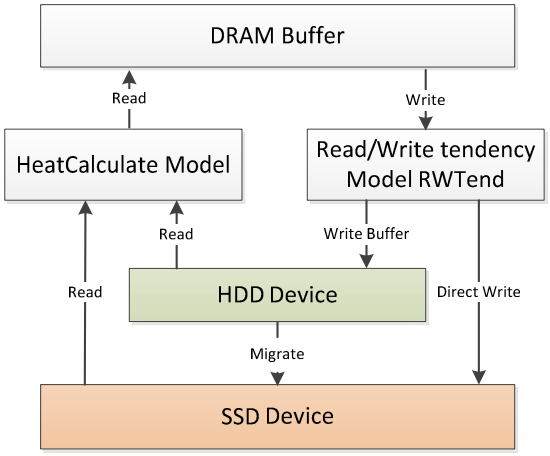


Fig. 2. HB-Storage Architecture

As shown in Fig.2, a migration not only changes the pages' location, but also updates SSD-index and HDD-index. The migration operation is shown in Algorithm 1. Line 2 handles the change of the page information. Line 3 makes sure the page 'evicted' from HDD is written to SSD.

Algorithm 1. MigratePage(Page *pg*)

1. **IF** $pg.placement = SSD$ **THEN**
2. $pg.placement \leftarrow HDD$
3. Insert *pg* into HDD index
4. **ELSE IF** $pg.placement = HDD$ **THEN**
5. $pg.placement \leftarrow SSD$
6. Remove *pg* from HDD index and update *pg* in SSD index
7. **END IF;**
8. set *pg* is dirty

3.2 HotnessCalculate Model

Different from the hot/cold technology used in DRAM of previous work, HB-Storage takes this technology to manage the HDD as buffer. Only the pages, proved as hot by

model HotnessCalculate, can be probably cached on HDD. Model HotnessCalculate sets a time threshold th to measure the hear degree, and defines some basic parameter:

- *hot access*: when the time interval between a physical access and the last one is less than t_h, this access is defined as hot access.
- *cold access*: when the time interval between a physical access and the last one is greater than t_h, this access is defined as cold access.
- *ordinary access*: if the access is not hot access or cold access, it is ordinary access.
- *hot state*: When the continuous N physical accesses are hot access, the page is regarded to in the hot state.
- *cold state*:When the continuous N physical accesses are cold access, the page is regarded to in the hot state.

As HotnessCalculate shows, one hot access or cold access cannot change the hotness state of a page. The reason is that if the hotness state can be modified by one access which may occur accidentally, the state switch between hot and cold would become very frequent, which is called 'state shake'. The 'state shake' may cause the page cached on HDD inappositely, which will reduce the overall read performance and increase unnecessary migration. To avoid the 'state shake', a rule is set that only when N(≥2) continuous physical accesses is hot/cold access, the hotness state is valid and believable. The greater N is, the stronger is the characteristics of page's hotness state. In this paper, N is set as 2.

Based on the definition as above, HotnessCalculate Model is designed as below.

Algorithm 2. HotnessCalculate(Page *pg*)

1. when *pg* is fetched from **SSD** at time *t*
2. **IF** *pg* is a new page **OR** $pg.t_1 = pg.t_2 = null$ **THEN**
3. $pg.t_1 \leftarrow t$ $pg.t_2 \leftarrow t$
4. $pg.heat \leftarrow cold$
5. **ELSE IF** $pg.t_1 = pg.t_2$ **THEN**
6. $pg.t_1 \leftarrow t$
7. $pg.heat \leftarrow cold$
8. **ELSE IF** $pg.t_1 \neq pg.t_2$ **THEN**
9. **IF** $t - pg.t_1 > t_h$ **AND** $pg.t_1 - pg.t_2 > t_h$
10. $pg.heat \longleftarrow cold$
11. $pg.t_2 \leftarrow pg.t_1$ $pg.t_1 \leftarrow t$
12. **ELSE IF** $t - pg.t_1 < t_h$ **AND** $pg.t_1 - pg.t_2 < t_h$
13. $pg.heat \longleftarrow hot$
14. $pg.t_2 \leftarrow pg.t_1$ $pg.t_1 \leftarrow t$
15. **ELSE**
16. *pg.hotness* has no change
17. $pg.t_2 \leftarrow pg.t_1$ $pg.t_1 \leftarrow t$
18. **END IF** ;
19. **END IF** ;
20. when *pg* is write buffer to **HDD**
21. $pg.t_1 \leftarrow null$ $pg.t_2 \leftarrow null$
22. set *pg* is dirty

Time threshold t_h is an important measure of the hotness state. As measured by number of pages, if the access interval is greater than HDD storage size, the page must not be cached on HDD. So t_h should be not more than HDD storage size. But HDD generally has a capacity advantage (larger than SSD) actually, which indicates t_h cannot be set as HDD storage size. On the other hand, the page access opportunities are mutually exclusive, if t_h is set as x‰, only the pages with top x‰ access frequency have a chance to become a hot state. This paper set t_h as 1‰ of SSD storage size.

Algorithm 2 set two parameters t_1 and t_2 to record the access time for page *pg*: t_1 is for last access time and t_2 is for the access time before last one. Line 9 indicates that if current access and last access are both cold access, the page becomes cold. Line 12 indicates that if current access and last access are both hot access, the page becomes hot. If both conditions as above do not appear, the page hotness state has no change, which shows two consecutive hot/cold accesses can affect the hotness state of page. So this policy can avoid the 'state shake' effectively. After the page is cached on HDD, its t_1 and t_2 are set as *null* as shown in line 20 and 21, which presents HotnessCalculat Model is not running for the pages on HDD. This algorithm is invoked when the page is fetched from SSD.

3.3 Read/Write Tendency Model RWTend

Because of SSD I/O performance asymmetry, the read-tendency pages should be stored on SSD while the write-tendency pages should be stored on HDD. RWTend model respectively calculate the SSD access cost and HDD access cost for the pages by the access statistics. This paper regards the pages having less SSD cost as read-tendency ones and the pages having greater HDD cost as write-tendency ones. For the HDD buffer, the accesses are all physical ones to disk, so this paper just counts the physical accesses for page. RWTend model defines some parameters as below.

- *Clean Access*: the page have no update in DRAM, it is 'clean' when evicted from DRAM.
- *Dirty Access*: the page is updated in DRAM, it is 'dirty' when evicted from DRAM.
- R_{phy}: counting the clean access.
- W_{phy}: counting the dirty access.
- CW_h: The cost of physical write to HDD.
- CR_h: The cost of physical read to HDD.
- CW_s: The cost of physical write to SSD.
- CR_s: The cost of physical read to SSD.
- *CMig*: the cost of migration from HDD to SSD.

Algorithm 3. RWTend(Page *pg*)

23. *CMig*←*CWs*
2. when *pg* is evicted from DRAM
3. **IF** pg is DirtyAcc **THEN**
4. $pg.W_{phy} \leftarrow pg.W_{phy} + 1$
5. **ELSE IF** pg is CleanAcc **THEN**
6. $pg.R_{phy} \leftarrow pg.R_{phy} + 1$
7. **ENDIF;**
8. $ssdcost \leftarrow pg.R_{phy} \cdot CR_s + pg.W_{phy} \cdot CW_s$
9. $hddcost \leftarrow pg.R_{phy} \cdot CR_h + pg.W_{phy} \cdot CW_h$
10. **IF** *ssdcost*−*hddcost* > *Cmig* **THEN**
11. **RETURN** *read-tend*
12. **ELSE IF** *hddcost*−*ssdcost*> *Cmig* **THEN**
13. **RETURN** *write-tend*
14. **ELSE**
15. **RETURN** *No-Tend*
16. when *pg* is write buffer to HDD **OR** *pg* is migrated to SSD from HDD
17. $pg.W_{phy} \leftarrow 0 \quad pg.R_{phy} \leftarrow 0$

RWTend model is shown in Algorithm 3. Because the page is cached on the HDD buffer only when it is evicted from SSD as 'dirty', the migration cost from SSD to HDD is void. When the page is migrated from HDD to SSD, the migration may cause some extra write operations to SSD if the page is evicted from DRAM as 'clean'. So CMig=CWs is set. Line 3 – 7 counts the physical access according to the page's dirty or clean. Line 8 – 9 calculates the SSD access cost and HDD access cost. Line 10 – 13 shows it is only when the difference between SSD access cost and HDD access cost is greater than CMig that the access load of the page has tendency. Line 14 – 15 shows if the difference between SSD access cost and HDD access cost is too little that the access tendency of the page is not clear, there is no migration. When the page location switches between SSD and HDD, Rphy and Wphy is set to 0, as shown in line 16 and 17. This algorithm is invoked when the page is evicted from DRAM.

3.4 Storage Management in HB-Storage

We discuss the storage management mechanism of the entire HB-Storage. Algorithm.4 shows the overall design of HB-Storage. We design SSD as the secondary storage device, and the pages are stored on SSD when first allocated. When the page was fetched from SSD by DRAM buffer, invoke the HotnessCalculate module to calculate the hotness state of the page. When the page was evicted from DRAM, invoke the RWTend module to calculate the read/write tendency of the page. If the hotness state of the page is hot and the page is write-tend, the evicted page from DRAM will be buffered on HDD; otherwise, the page will be directly written to SSD.

Algorithm.4. HB-Storage(Page *pg*)

```
1.  IF pg is fetched from SSD
2.     HotnessCalculate(pg);
3.  END IF;
1.  IF pg is evicted from DRAM
2.     IF pg.placement = SSD AND pg.hotness=hot THEN
3.        IF RWTend(pg)=write-tend AND pg is DirtyAcc THEN
4.              pg is write on HDD to buffer
5.           MigratePage(pg);
6.        ELSE
7.           pg is direct write to SSD
8.        ENDIF;
9.     ELSE IF pg.placement = HDD THEN
10.       IF RWTend(pg)=read-tend THEN
11.          pg is direct write to SSD
12.          MigratePage(pg);
13.       ELSE
14.          pg is write on HDD to buffer
15.       END IF;
16.    END IF;
17. END IF;
```

In Algorithm.4, if the page on SSD is hot and write-tend, it will be writtend on HDD to buffer (line 4 to 7 in Algorithm.4). However, it must ensures that the access to page is DirtyAcc. If the access to page is CleanAcc, the write operation to HDD will increase an additional physical write opeartion to system. We no longer calculate the hotness state of page buffered on HDD. Once the page on HDD becomes write-tend, it will be migrated from HDD to store on SSD.

HB-Storage invokes the *HotnessCalculate* module to calculate the hotness state of page, and it invokes the *RWTend* module to calculate the read/write tendency of page. The pages found to be frequently accessed and write-tend will be buffered on HDD. Although the physical read performance of these pages are realtively poor, these pages can effectively reduce the cost of frequently write operations of them on SSD, which as a result can improve the performance of entire system.

4 Performance Evaluation

4.1 Experiment Setup and Datasets

The experimental system can be divided into three modules: the storage manager, the DRAM buffer and the HB-storage manager. Besides, it applies the B+-tree to index the pages storing data. The buffer manager is consisted of free space manager and non-free space manager. The DRAM buffer is managed by the LRU algorithm. Both of them are combined with the HB-storage manager, which completes the management of the hybrid storage system by implementing the *HotnessCalculate* and

RWTend module. The experiment is implemented in C++ and ran on Debian GNU/Linux 2.6.21 operating system, where the size per page is 4KB. The buffer size is set to be 4MB. The experiment is ran on real devices, including the plain HDD and the low-end SSD(OCZSSD2-1C64G).

Table 1 describes the actual read/write latency of the HDD and OCZSSD2-1C64G SSD devices, and gives the read/write performance comparison between HDD and SSD, where the cost value is normalized by multiples of the minimal cost. The larger the read/write latency, the faster the read/write speed, and the greater the read/write cost. i.e., the OCZSSD2-1C64G SSD read is 23 times faster than HDD read, while the HDD write is 20 times faster than OCZSSD2-1C64G SSD write. The read/write performance comparison between SSD and HDD is apparent, and the I/O asymmetry of SSD is very obvious.

Table 1. R/W Performance Comparison Between HDD and SSD

R/W Type	Latency(ms)	IOPS	Cost
SSD read	0.482	2057	1
SSD write	117.89	8.48	243.5
HDD read	11.0953	90.11	23
HDD write	5.86	170.7	12

The I/O cost parameter of the RWTrend algorithm can apply the cost value in Table 1, e.g., we have parameters CW_h=12, CR_h=23, CW_s=243.5, CR_s=1 for the hybrid storage system with HDD and the OCZSSD2-1C64G SSD. In the experiment, we record the number of physical read/write operations in the running process of HB-Storage, which can provide the basis for the calculation of HB-Storage. Besides, we give an analysis of the number of physical read/write operations subsequently.

Table 2. Description of Experimental Dataset

Trace Type	Request Num	Page Num	R/W proportion	Access Locality
T9182	300000	10000	90%/10%	80%/20%
T5582	300000	10000	50%/50%	80%/20%
T1982	300000	10000	10%/90%	80%/20%
T9155	300000	10000	90%/10%	50%/50%
T5555	300000	10000	50%/50%	50%/50%
T1955	300000	10000	10%/90%	50%/50%

We generate the experimental datasets with the open source tool DiskSim [12]. As Table 2 describes, the dataset consists of trace six files, which include records with different read/write proportion and access locality respectively. With this diverse dataset, we can test the performance of HB-Storage under different read/write performance and different access locality, in order to get the more universal and persuasive experimental results.

The initial experimental setup is to assume that all the pages are stored on SSD, and the HDD is empty. In the experiment, we measure the effectiveness of HB-Storage according to the HDD cache performance.

4.2 Experimental Results and Analysis

In order to weigh the effectiveness of HDD write cache, in the experiment we set two testing factors, one of which is the write hit ratio *WH* of HDD cache, and another is the read hit ratio *RH* of HDD cache, e.g., assume that there were *N* write requests and *M* read requests in the experiment, while there were N_H write requests and M_H read requests happened on HDD, then the testing factors *WH* and *RH* can be calculated as formula (1):

$$WH = \frac{N_H}{N} \qquad RH = \frac{M_H}{M} \tag{1}$$

The higher the *WH* value is, the more the number of write operations on HDD write cache is, and the more the write cost on SSD reduces. Similarly, the lower the *RH* value is, the more the number of read operations on SSD is, and the more the read cost on HDD reduces. If the *WH* value is high while the *RH* value is low, the HDD write cache is most effective at this time. As one write operation on HDD should correspond to one read operation at least in the experiment, if it can be achieved that several write operations on HDD correspond to one read operation in average with HB-Storage, we can ensure that HB-Storage works effectively, when the *WH* value is apparently larger than the *RH* value. We records the total number of read/write requests and the read/write operations happened on HDD in the experiment, and then we can calculate the *WH* and the *RH* value of HDD cache.

As Figure 3 shows, as the proportion of read operations in the trace file gradually increases, the disparity between the *WH* value and *RH* value HDD write cache in HB-Storage is more and more large, and it reflects that the HDD write cache in HB-Storage becomes more and more effective. There are so many read requests in the trace file T9155 and T9182 that only a few pages can be cached on HDD, and the

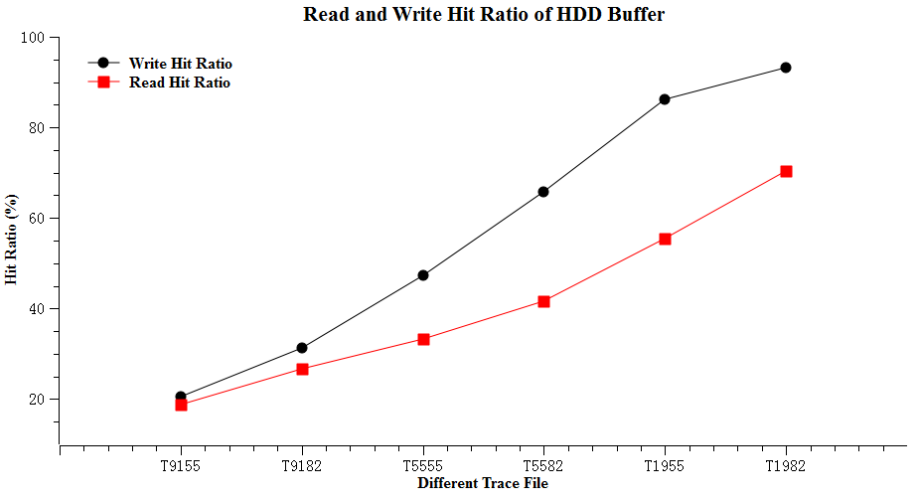


Fig. 3. Read/Write Hit Ratio of HDD Cache

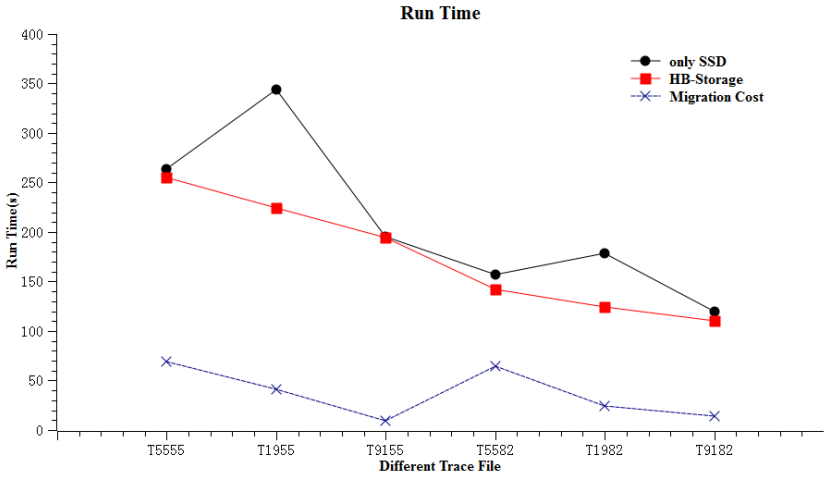


Fig. 4. Comparison of Run Time between SSD and HB-Storage

write requests to the cached pages on HDD are so few that almost one write operation correspond to one read operation, so the *WH* value and *RH* value are very close to each other in the trace file T9155 and T9182.

Besides, there are so many write operations in the trace file T1955 and T1982 that HB-Storage can effectively find the write-tend and hot page to cache on HDD. The results illustrate that HB-Storage suits well for improving the performance of low-end SSD on write-intensive workloads.

We record the run time of the dataset on low-end SSD and HB-Storage respectively, which can highlight the effect of HDD write cache in HB-Storage. We not only record the run time on pure SSD and HB-Storage, but also record the additional latency due to the migration operation happened in HB-Storage. As Figure 4 shows, the design of HDD write cache in HB-Storage can significantly improve the write performance of SSD in the write-intensive workloads, and HB-Storage shows better performance than pure SSD overall. Besides, the migration cost maintains a relatively low value, which illustrate that the buffering operation and the migration operation of HB-Storage are both accurate.

5 Conclusions

In this paper, we proposed an effective hybrid storage system called HB-Storage to cope with the deficiency of low-end SSDs by using a HDD write buffer. HB-Storage utilized both the techniques of access hotness and the analysis of read/write tendency to introduce an HDD as an effective write cache, thereby improved the overall performance of the storage system.

In the future, we will carry out research on other hybrid storage architectures. On the other side, as the price of SSDs is high for a long time, we will combine with cost factors to conduct our research, while giving more attention on high performance/cost

ratio. We will also consider more creative storage management solution in order to achieve efficient storage performance with low-cost devices.

Acknowledgement. This paper is supported by the National Science Foundation of China (No. 60833005 and No. 61073039).

References

1. Agrawal, N., Prabhakaran, V., Wobber, T., et al.: Design Tradeoffs for SSD Performance. In: Proceedings of USENIX 2008 Annual Technical Conference on Annual Technical Conference, pp. 57–70 (2008)
2. Kgil, T., Roberts, D., Mudge, T.: Improving NAND Flash Based Disk Caches. In: Proceedings of Computer Architecture (ISCA 2008), Beijing, China, pp. 327–338 (2008)
3. Kgil, T., Mudge, T.: Flashcache: A NAND Flash Memory File Cache for Low Power Web Servers. In: Proceedings of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems, New York, USA, pp. 103–112 (2006)
4. Yang, P., Jin, P., Yue, L.: A Time-Sensitive and Efficient Hybrid Storage Model Involving SSD and HDD. Chinese Journal of Computers 35(11), 2294–2305 (2012)
5. Koltsidas, I., Viglas, S.D.: Flashing up the Storage Layer. Proceedings of the VLDB Endowment 1(1), 514–525 (2008)
6. Ou, Y., Härder, T., Jin, P.: CFDC: A flash-aware buffer management algorithm for database systems. In: Catania, B., Ivanović, M., Thalheim, B. (eds.) ADBIS 2010. LNCS, vol. 6295, pp. 435–449. Springer, Heidelberg (2010)
7. Ou, Y., Härder, T., Jin, P.: CFDC: A Flash-Aware Replacement Policy for Database Buffer Management. In: Proceedings of the Fifth International Workshop on Data Management on New Hardware, New York, USA, pp. 15–20 (2009)
8. Li, Z., Jin, P., Su, X., Cui, K., Yue, L.: CCFLRU: A New Buffer Replacement Algorithm for Flash Memory. IEEE Transactions on Consumer Electronics 55(3), 1351–1359 (2009)
9. Jin, P., Ou, Y., Härder, T., Li, Z.: AD-LRU: An efficient buffer replacement algorithm for flash-based databases. Data & Knowledge Engineering 72, 83–102 (2012)
10. Cheong, S.-K., Jeong, J.-J., Jeong, Y.-W., Ko, D.-S., Lee, Y.-H.: Research on the I/O Performance Advancement of a Low Speed HDD Using DDR-SSD. In: Park, J.J., Yang, L.T., Lee, C. (eds.) FutureTech 2011, Part I. CCIS, vol. 184, pp. 508–513. Springer, Heidelberg (2011)
11. Canim, M., Mihaila, G.A., Bhattacharjee, B., Ross, K.A., Lang, C.A.: SSD Buffer Pool Extensions for Database Systems. Proceedings of the VLDB Endowment 3(1-2), 1435–1446 (2010)
12. `http://www.pdl.cmu.edu/DiskSim/` (accessed in March 2013)

An Efficient Strategy of Building Distributed Index Based on Lucene

Tiangang Zhu[1,4], Yuanchun Zhou[1], Yang Zhang[1],
Zhenghua Xue[2], Jiwu Bai[3], and Jianhui Li[1]

[1] Computer Network Information Center, Chinese Academy of Sciences, China
[2] Chanjet Information Technology Co. Ltd., China
[3] Jiyuan Power Supply Company of Henan Electric Power Company, Henan, China
[4] University of Chinese Academy of Sciences, China
{zhu,zyc,zhangyang,zhxue,lijh}@cnic.cn

Abstract. With the arrival of big data era, the increasing scale of data available poses a great challenge to industry and academia. Efficient query and retrieval of large amount of data becomes more and more necessary. In this paper, we propose an efficient and smooth strategy of building distributed index for large amount of text. In order to improve memory usage and less manual intervention, the proposed strategy uses dynamic threshold setting other than static threshold. Dynamic threshold setting can also avoid Out of Memory(OOM) issue. For the purpose of loading balance, we also design a novel MinHeapPartition strategy to replace the default HashPartition. Because of continuous sending the intermediate data to the reducer with the lowest loading, the MinHeapPartition strategy can maximally make sure each reducer process approximately equal data loading. To validate the proposed strategy in efficiency and scalability, we build a distributed index based on Apache Hadoop and Lucene open source framework. In our experiment, we successfully index up to 1.02TB text data. Experiment results show that our strategy achieves 20% performance improvement.

Keywords: MapReduce Hadoop Lucene distributed index.

1 Introduction

In this age of data explosion, people's expectation for searching effective information from a massive volume of data grows stronger and stronger. Inverted index [1], which is an index data structure storing a mapping from content to its locations, is a common and useful approach to allow fast full text searches. Lucene, which was originally written by Doug Cutting in 1999, has been widely recognized for its utility in full text indexing and search capability. With the increase of the amount of the text data, a distributed method Hadoop is cited to deal with the efficiency problem. Apache Hadoop , derived from Googles MapReduce [2] and Google File System [3] papers, is an open-source software framework that supports data-intensive distributed applications.Apache provide

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 40–45, 2013.

a strategy using Hadoop and Lucene to build distributed index and store the index on Hadoop Distributed File System(HDFS) while the text data is massive. Although the strategy is feasible, there still exists much room to improve on the efficiency of building distributed index.

In this paper, we present new strategies which can cope with the large-scale text data more robust and more efficiently while building distributed index.Experiment results show that our new strategy achieves 20% improvement while the text data being indexed is up to 1.02TB.

The rest of the work is organized as follows: in Section 2, we introduce the main points of the Apaches strategy for building distributed index . Section 3 describes our strategy for improve the memory usage. In the Section 4, we presents the strategy to optimize the loading balance while building distributed index. Finally, in Section 5, we summarize our gains.

2 Apache's Strategy for Building Distributed Index

Apache provides a strategy to build distributed index, which gives a utility to build or update a distributed index using Lucene and Hadoop. The most basic three steps in Lucene are as follows:

Step 1: Adding fields to a document
Step 2: Adding document to an index
Step 3: Optimize the index

Hadoop implements a computational paradigm named map/reduce. In Apache's strategy, the map phase formats, analyzes and parses the input data, finishes steps "Adding fields to a document" and "Adding document to an index" in parallel, and the reduce phase collects the updates to each Lucene instance, finishes the step "Optimize the index" again in parallel.

As an optimization, Apache's strategy uses the "Combiner" which can implement local aggregation. It cuts down the amount of the intermediate data transferred from mappers to reducers. Besides, the "Partitioner" is also used to partition the intermediate data from mappers. The default "Partitioner" is "HashPartitioner" which computes the hash value of the key modulo the number of reduce tasks, and it will divide up the intermediate key space roughly evenly.

The part of building distributed index above-mentioned is the main content of the Apache's strategy. Our new strategy mainly modifies the "Combiner" and "Partitioner" to improve the memory usage and load balance, so that to improve the efficiency of building distributed index.

3 Dynamic Threshold Strategy

The Combiner in Apache's strategy combine multiple intermediate forms into one intermediate form. More specifically, the input intermediate forms are single-document RAM indexes, and an output intermediate form contains a multi-document RAM index. All the RAM indexes will be load into memory before

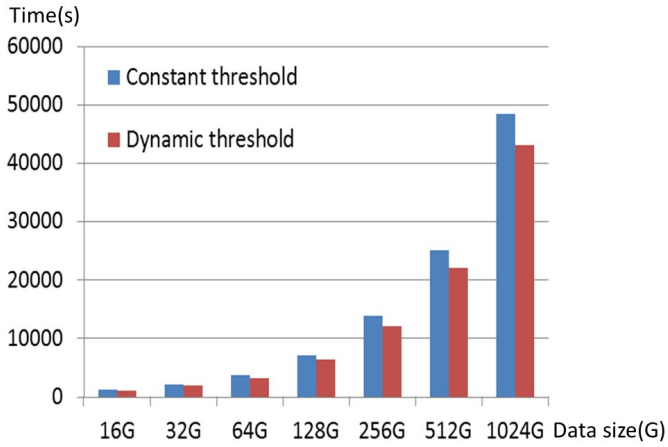


Fig. 1. Compared to constant threshold, dynamic threshold strategy improve performance by 10%

Combiner runs. Therefore, the larger the scale of input data, the larger the size of RAM indexes will be loaded into memory. While Combiner is running, the exception OOM will occur.

Apache's strategy provides a method: restricting the scale of the input intermediate forms of Combiner. The root cause of the OOM is that the scale of input intermediate forms of one Combiner is so large that many RAM indexes are loaded into memory, so the original strategy sets a threshold for Combiner to solve this problem. If the input intermediate forms are too large, only part of the RAM indexes whose size is under the threshold will be loaded into memory and processed, and the rests will be loaded and processed next time.We use X = (X_1, X_2,...,X_n) represents the input data, and use function F_p represents the processing of loading data into memory in Combiner. T represents the threshold. The procedures of Combiner are as follow:

```
/*Procedure of Combiner*/
if size(X) < T
 Fp(X)
else
 Fp(partOf(X))
```

However,the original strategy failed to consider this relationship between the size of input data and the memory, and the threshold here is constant. If the threshold is too small, we will not make full use of the memory;if the threshold is so large to exceed the memory size, the OOM will occur. An appropriate threshold is the key to improve the data processing efficiency. In this paper, we propose a dynamic threshold strategy which can dynamically select a suitable threshold to avoid the OOM issue. The threshold is determined by comprehensively considering the relationship between the current available size of memory rather than setting a

constant threshold. The threshold will be large if the current available memory size is large. Our Dynamic threshold strategy is as follow:

```
/*Procedure of newCombiner*/
T = current available memory size;
if size(X) < T
 Fp(X)
else
 Fp(partOf(X))
```

Our Dynamic threshold strategy can adjust the data size input into Combiner according to the current available memory size. Fig.1 shows the results of experiments in which the input text data size is from 16G to 1.02 TB. Because of making full use of memory, it can improve the efficiency of building distributed index about 10% and avoid OOM.

4 Minimum Heap Partition Strategy

The partitioner is responsible for scheduling the output key/value pairs from mappers to reducers. When a job is configured to have a reduce phase, the output from map phase will be split into partitions (one partition per reduce task). The default partitioning strategy , HashPartition, uses the hash code of the key modulus the number of reducers, key.hashCode() % conf. getNumReduceTasks(). If the job,for example, has three reducer, the default partition for a key will be key.hashCode() % 3.

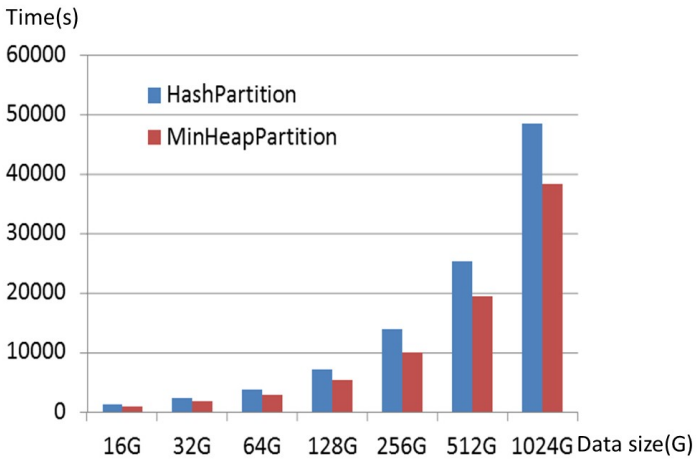


Fig. 2. Runing Times Contrast of HashParitition and MinHeapPartition. The Efficiency of MinHeapPartition Can Be Increased by 20%.

The HashPartition simple and easy to implement, and it is effective for performance to ensure that each reducers receive key/value pairs with approximately same quantity. However, the strategy takes no account of the size of key/value pairs. Before being sent to Mapper, the input data will be split in pieces using the character "\n" as a separator. The input data is in text format,and the contents between two "\n" can be lot and also can be not. If using the HashPartition, the default partitioning strategy, each of reducers will receive the same quantity of key/value pairs, but the size of each reducers gain may not be the same. Some reducers which receive larger size of data will finish their tasks slowly, but others will finish their tasks more quickly, and the resources will be wasted in generally speaking. In other words, the efficiency of the program will be lowered.

To address the problems above, We create a new partitioning strategy called "MinHeapPartition" which can make each reducer receive data with the same size. The "MinHeapParition" consists of three steps as follow:

Step 1: Getting the size information of each input key/value pairs. We can get the value size in the Map phase.

Step 2: Building a min-heap. The number of heap's nodes is equals to the number of reduce tasks. Each of nodes store the reducer number and the data size received by the reducer. Step 3: Sending the new key/value pair to the reducer whose data size is the least in the minimum heap, and refresh the minimum heap.

The new MinHeapPartition strategy can adjust the size each reduce tasks get and change "the number of key/value pairs balancing" to "size of key/value pairs balancing", which is more efficient. Fig.2 presents the results of experiments in which the input text data size is from 16G to 1.02T. Because of load balance on each reducer, our MinHeapPartition strategy can make full use of computing resources on each reducer, so it that maximum the processing capacity of the program. The results show that the efficiency of building distributed index can be improved up to 20%.

5 Conclusion

In this paper, we propose an efficient strategy for building distributed index for large scale of text data. To address the out of memory issue occured in the process of index building, we uses the dynamic threshold strategy other than the static threshold to improve memory usage and less manual intervention. In addition, we also design a novel MinHeapPartition stragegy to replace the default HashPartition strategy. The MinHeapPartition strategy make sure that each reducer process approximately equal data loading.

We perform various experiments with up to 1.02TB text data to validate the proposed strategy in efficiency and scalability. Experiment results show that our proposed novel strategy outperforms the default strategy in building the distributed index. The performance improvement even achieves 20%.

In the future, we will continue to explore more sophisticated strategy that can improve the performance of building distributed index on large scale of data. Maybe incorporating the compression technique will be a promising direction.

Acknowledgements. Yuanchun Zhou is the corresponding author. This work is partially supported by Natural Science Foundation of China under Grant No. 61003138 and 91224006, the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA06010202, XDA05150401 and XDA05050601-4, "Twelfth Five-Year" Plan for Science & Technology Support under Grant No. 2012BAK17B01-1.

References

1. Ribeiro, de Arajo Neto, B., Baeza-Yates, R.: Modern information retrieval, p. 192. Addison-Wesley Longman, Reading (1999) ISBN 0-201-39829-X
2. Dean, J., Ghemawat, S.: MapReduce: Simplified data processing on large clusters. In: Proceedings of the 6th Symposium on Operating System Design and implementation (OSDI 2004), San Francisco, California, pp. 137–150 (2004)
3. Ghemawat, S., Gobioff, H., Leung, S.-T.: The Google File System. In: Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003), Bolton Landing, New York, pp. 29–43 (2003)
4. Lin, J., Schatz, M.: Design patterns for efficient graph algorithms in MapReduce. In: Proceedings of the Eighth Workshop on Mining and Learning with Graphs, pp. 78–85. ACM, New York (2010)
5. Gufler, B.: Load Balancing in MapReduce Based on Scalable Cardinality Estimates. Data Engineering (ICDE). In: 2012 IEEE 28th International Conference, pp. 522–533. IEEE Press, Washington, DC (2012)
6. Chen, Z., Zhu, C., Cheng, W., Song, Q., Cai, S.: Research of distributed index based on lucene. In: Jin, D., Lin, S. (eds.) Advances in EECM Vol. 1. LNEE, vol. 139, pp. 115–121. Springer, Heidelberg (2012)
7. Jiang, D.: The performance of MapReduce: An in-depth survey. Proceedings of the VLDB Endowment, 472–483 (2012)
8. Jiang, D., et al.: Map-join-reduce: Towards scalable and efficient data analysis on large clusters. IEEE Transactions on Knowledge and Data Engineering (2010)
9. Babu, S.: Towards automatic optimization of mapreduce programs. In: Proceedings of the 1st ACM Symposium on Cloud Computing, pp. 137–142. ACM, New York (2010)
10. Chaiken, R., Jenkins, B., Larson, P.-A., Ramsey, B., Shakib, D., Weaver, S., Zhou, J.: Scope: easy and ecient parallel processing of massive data sets. PVLDB 1(2), 1265–1276 (2008)
11. Justin, Z., Moffat, A.: Inverted Files for Text Search Engines. ACM Computing Surveys 38(2), 1–56 (2006)

Adaptive Sequential Prefetching for Multiple Streams

Yong Li, Dan Feng, Zhan Shi, and Qing Liu

School of Computer, Huazhong University of Science and Technology,
Division of Data Storage System, Wuhan National Lab for Optoelectronics,
Wuhan, China, 430074
li.yong.xyz@gmail.com, {dfeng,zshi,qing}@hust.edu.cn

Abstract. Modern storage systems are becoming increasingly consolidated. As a result, there exists a competition for resources among concurrent streams. Sequential prefetching is widely used in modern storage system. Most of them always ignore the impact of diversified access rate of concurrent streams. However the concurrent streams with diversified access rate will introduce several problems. The streams with fast access rate may evict the cache blocks of the streams with slow access rate and lead the slow streams re-fetch the evicted cache blocks in future access, which makes the prefetching wastage and unfairness to the slow streams. We design and implement a novel, adaptive algorithm, named ASPM to solve these problems. Our experiments show that, compared with LRU and AMP, ASPM can achieve significantly improvement in fairness among concurrent streams and slightly in performance (average on 6.8%).

Keywords: sequential prefetching, adaptive, storage system.

1 Introduction

In modern storage systems, the speed of processor is much faster than the storage devices. So, the processor has to waited huge number of cycles for the completion of disk I/Os. A large amount of literatures has researched to hide the I/O latency. Caching is a fundamental technique in these literatures, like LRU [1], LRU-K [2, 3], LIRS [4], MQ [5]. Most of them exploit the locality of the workloads to significantly reduces the number of disk I/O and improve the performance.

Prefetching is another important technique widely used in storage systems. It exploits spatial localities in workloads and merges several disk I/O operations into one big continuous disk I/O operations. Prefetching will prefetch un-accesses blocks from disk to cache whose address is close to the last accessed block. A prediction scheme is always used in prefetching to mine the block access correlations, such as C-Miner [6]. Determining appropriate prefetching degree is also very important for prefetching mechanisms. There are several researches focus on learning the sequential behavior of workloads and determine appropriate prefetching degree, such as SARC [7] and AMP [8]. The accuracy of prefetching is very important. The prefetching can reduces the disk I/O operation if

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 46–55, 2013.

the prefetched blocks hit in future accesses and gain performance improvement. However if the prefetching blocks are un-accessed until them are evicted from cache, it will decrease the performance.

Common approaches are rate-oblivious and means that the prefetching does not consider the diversified access rate of concurrent streams. However, it is very common in practice that there exist concurrent streams with diverse access rate, for example video player, big file copy work with mp3 listening and pdf reading. The concurrent streams with diverse access rate will introduce performance degradation and unfairness for the streams with slow access rates. Because fast streams may need too much cache blocks and evict the cache blocks of slow streams even if the slow stream have better sequential access pattern. It may lead the slow streams re-fetch the evicted cache blocks in future, which make prefetching cache wastage and performance degradation. Obviously, the behavior may bring the unfairness between fast and slow streams. In this paper, we explore the integration of rate-aware scheme into the adaptive sequential prefetching algorithms. Our simulation study shows that our approach can significantly improve the fairness between heterogenous streams and as well as slight performance improvement.

The rest of the paper is organized as follows. Section 2 presents the design of our algorithms. Section 3 presents the experimental evaluation. Section 4 presents the relate works. Section 5 presents the conclusion.

2 Adaptive Sequential Prefetching for Multiple Streams

This section provides the detail description of our algorithm. First we outline the unfairness among concurrent streams with diverse access rate. Then we describe the design of adaptive prefetching degree. And last we describe the design of prefetching resources allocation to streams.

2.1 Unfairness Problem

We theoretically analyze the un-fairness problem when multiple sequential streams shared in cache. Let the cache size is C and the number of streams is N. Each stream has an average access rate r_i, average request size s_i and prefetching degree p_i $(i < N)$. If the sequential request hit in the previous prefetching data, it will not performs disk I/O. Otherwise it incur an I/O miss and perform synchronous prefetch with p_i blocks. For asynchronous prefetching, it set one of prefetching blocks as trigger block. If the trigger block is accessed, it will perform the asynchronous prefetching with p_i blocks. So the stream always have least p_i prefetched blocks in the cache. The blocks arriving rate of stream i can be expressed as $r_i \times s_i + p_i$. Let T_i be the average requests arrive interval of stream i. T_i is the reciprocal of the request arrive rate and equals $T_i = 1/r_i$. Let L be the average life of a block in the cache. L can be computed through

the cache size divide by the total blocks of all streams, as follow:

$$L = \frac{C}{\sum_{i=1}^{N} r_i \times s_i + p_i} \tag{1}$$

The L denote how long a block will be evicted from cache. If the interval time of a stream is smaller than the L mean that the prefetched blocks will be evicted from cache with high probability before them are accessed. So, we get the conclusions that the fast stream can degrade the performance of slow streams. Especially at the burst access of the fast stream, the performance degradation will increase rapidly. Intuitively, giving more prefetching resources to the slow stream can alleviate this performance degradation.

2.2 Adaptive Prefetching Degree

The key idea in the prefetching algorithm is how to realize and match the variation of the workload. It can be simply described by "how much to prefetch" and "when to to prefetch". In our scheme we use two parameters to describe the necessity of adjusting prefetching degree for streams: the sequential miss and the access rate. If the arrived request incur a sequential miss, it can get the conclusions that the previous prefetching degree is too conservative. This is for two reasons. One is that the setting of previous prefetching degree is just too small and the other is that the rate of the stream has become faster. In order to avoid the sequential miss for next requests, our algorithm multiply the degree with a const value M to perform more aggressive prefetching. As a result, the prefetching degree will be adaptive to the change of access rate of streams, which can effectively avoid the wastage of the prefetching cache.

The streams with diverse access rates have different utility in performance for the same size of prefetching cache. For example, fast stream may not have sufficiency prefetched blocks to catch up the increased load, while slow streams may have many unconsumed blocks. Our algorithm monitor the average life of blocks for streams in each fixed interval. If the average life of a stream is larger than the threshold T, our algorithm will shrink the prefetching degree with a const value S. The setting of S is directly proportional to the access rate of streams.

We call multiply the prefetching degree to multiplicative increase and call shrinking the prefetching degree with a const value to linear decrease. This idea is similar to the windows concept used in TCP congestion control. Our experiments show that the adaptive prefetching degree can better catch up the variety of the workloads than static method. If the workload became more active, the prefetching degree can increase rapidly. In contrast, if the workload became quiet, the prefetching degree will drop to a suitable value in short time.

2.3 Adaptive Partition Size

The goal of our algorithm is aim to maximum the hit ratio of prefetching cache. The streams with diverse access rate will obtain different hit ratio under identify

prefetching cache size. Our algorithm use the marginal utility of the hit ratio as the metric of the adjustment. The marginal utility for an stream is the increase in hit ratio that will be seen by this stream if the cache size increases by a single block, which is defined as:

$$MU(n) = h(n+1) - h(n) \tag{2}$$

where n refers the cache size and h(n) refers to the hit ratio. The relationship between the cache size and the hit ratio will be not linear once the cache size beyond a Threshold, which is also proved by [7,16]. When the size of prefetching cache is small than T, it will significantly increase the sequentially hit ratio with any additional cache blocks. However if the size exceed the threshold T, any increase of the prefetching size is not possible to drive the number of sequentially misses to zero and the sequentially hit ratio maintains a stable level. Our algorithm is a partition-based cache management scheme and maintain appropriate-sized partitions for each stream. For each partition, we divide the partition into two segments with diverse marginal utility, show as Figure 1. The next-access segment include the blocks which are prefetched for future access and not accessed. The prev-access segment include the blocks which are already accessed by previous request. Obviously, next-access segments are more useful than "prev-access" segments for future request. So, we give larger weight to next-access segments than prev-access segment.

When a demand access misses in the cache and the partition does not have any free cache block left. A cache block has to be taken away from one of the partition. The partition with smallest marginal utility will be chosen as victim. Then, we apply commonly used cache replacement policy to evict a cache block for demand access. To avoid scan all segments every time, we maintain a variables MU_{min} that refers to the minimum marginal utility of a partition. Once a stream change its marginal utility, our algorithm update the MU_{min} if the new marginal utility is smaller than MU_{min}. Our cache management algorithm is show as in Algorithm 1.

3 Performance Evaluation

we use trace-driven simulations to evaluate the performance of the ASPM.

3.1 Experiment Setting

We built a simulator that implements the ASPM, AMP [8], and LRU algorithms [1]. We also interfaced the Disksim 4.0 [10] to simulate the disk behaviors, which is an accurate disk simulator. The disk drive we modeled is the ibm18es with 7200 RPM and 9GB capacity. Its maximum read/write seek time is 12.8ms/13.0ms, and its average rotation time is 3.0ms.

We use IOmeter [11] to generate the workload and blktrace [12] to capture the trace. First we set the characteristic of the workload in IOmeter. Then we run this workload on a host machine and use blktrace to capture the I/O trace. At last we use this trace to this trace to drive the simulator.

Algorithm 1. shows the pseudo-code for our cache management algorithm.

Initilize: Total number of partitions is n. The size of request windows is WS. Set $life\_count_i = 0$, $prefetch\_degree_i = 0$ and $window\_count = 0$

block x is request:

```
1: window_count++
2: if window_count > WS then
3:    adjustPrefetchDegree()
4: end if
   in the case of hit:
5: moveToMRU()
6: if x is the trigger block then
7:    done asynchronous prefetch
8: end if
   in the case of miss:
9: selectVictim() ← partition_i
10: deleteTailBlock(partition_i) ← x
11: if detecting sequential access then
12:    done synchronous prefetch
13: end if
14: if evicted block is prefetching block and have not be accessed then
15:    life_count_i ++
16: end if
17: if prevBlock(x) in cache then
18:    life_count_i --
19: end if
   adjustPrefetchDegree():
20: for i = 1 to n do
21:    if life_count_i > 0 then
22:       prefetch_degree_i = prefetch_degree_i − S
23:    else
24:       prefetch_degree_i = prefetch_degree_i * M
25:    end if
26: end for
```

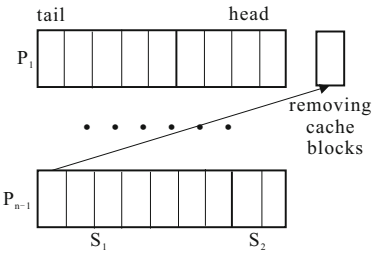


Fig. 1. Each stream's partition is broken into two segments, "prev-access" segment denote S_1 and "next-access" segment denote S_2. "Prev-access" segments contain already accessed cache blocks and "next-access" segments contain un-accessed cache blocks. Our algorithm will removing the cache block with low MU value from the victim partition to the demand partition.

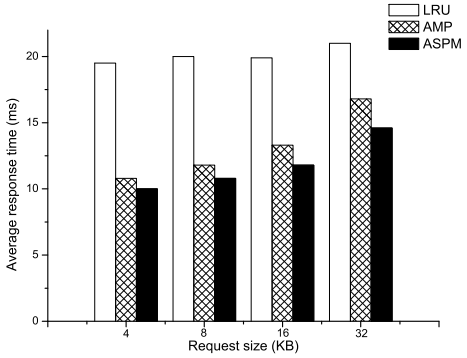


Fig. 2. Concurrent streams with varying request size

3.2 Concurrent Streams with Varied Request Size

As discussed above sections, our algorithm focus on the prefetching under concurrent streams. First we use synthetic workloads to evaluation the performance of LRU, AMP, ASPM under different access rates. The synthetic workloads used in all experiments have total 100 streams, include 20 random streams, 20 fast stream, 30 slow streams and 30 medium rate stream. The rate proportion of four kind streams is 100 : 100 : 10 : 1.

To evaluate how the LRU, AMP, ASPM algorithms responds to different requests size. We generate a set of trace with different request size which vary from 4KB to 32KB. Figure 2 shows the results in terms of the average request response time with varied request size.

The performance of LRU algorithm is lower than the AMP and ASPM algorithms. AMP and ASPM perform disk I/O in batch fashion with help of prefetching. So, it can avoid most of thrashing seek introduce by the interleaved access of the concurrent streams. Compared with LRU algorithm, AMP algorithm improved about average of 34.7% on performance and ASPM algorithm improved about 41.5% on performance. The ASPM algorithm outperforms the AMP algorithm about 6.8%. Especially, in the case of request size 32KB, AMP algorithm only improved about 20% compared with LRU. However, the improvement of our algorithm is about 30.5%, which outperform the AMP algorithm about 10.5%. The average improvement on performance is not very large for both AMP and ASPM. Recall that we focus on how to balance the prefetching cache between fast and slow streams. The access rate of slow streams are much smaller than fast streams, which mean that the percentage of load for slow streams is very small. So, there is a inherent limit to further improvement of performance for AMP and ASPM.

3.3 Concurrent Streams with Varied the Percentage of Streams

This section we evaluate the performance of AMP algorithm and ASPM algorithm with vary the percentage of slow streams number. Intuitively, more large

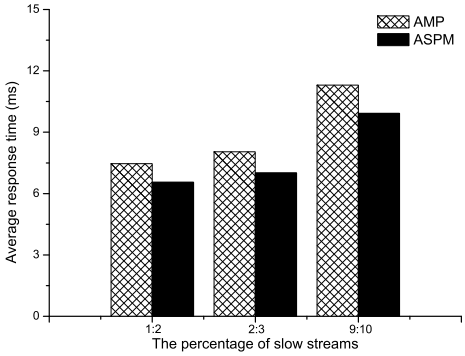


Fig. 3. The average request time of AMP algorithm and ASMP algorithm with growing the percentage of slow streams numbers

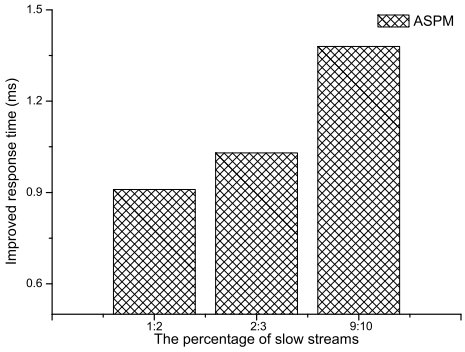


Fig. 4. The ASPM algorithm's improved performance over AMP algorithm with growing the percentage of slow streams numbers

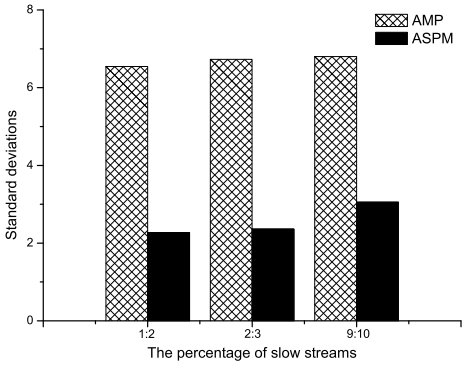


Fig. 5. The standard deviations of AMP algorithm and ASMP with growing the percentage of slow streams numbers

percentage slow streams have will gain more performance improved. We generate three set of workloads to use in experiments. The percentage of slow streams is 50%, 66.6% and 90% respectively. The average request size is fixed at 16KB. Figure 3 shows the results in terms of the average request response time with growing the percentage of slow streams numbers. Figure 4 shows the ASPM improvement performance over AMP with growing the percentage of slow streams numbers.

As expected, ASPM algorihtm does better with increase the percentage of slow streams. ASPM algorithm improved 0.91ms of average response time when the percentage is 50% and improved 1.03ms of average response time when the percentage is 66.6%. When the percentage of slow streams improved to the 90%, the ASPM algorithm further up to obtain 1.38ms of average response time improvement.

3.4 Fairness

This section we evaluate standard deviations of all streams for AMP algorithm and ASPM algorithm with vary the percentage of slow streams number. We use the previous workloads to drive the experiments. The standard deviations between the streams is good to denote the fairness among the streams. Figure 5 shows the results in terms of the average request response time with growing the percentage of slow streams numbers.

The lower standard deviation denotes better result of fairness. Opposite, the larger standard deviations mean that it obtains the worse result of fairness. As the experiments show, the ASPM algorithm obtains average 2.56 of standard deviation. However, the AMP algorithm obtains about average 6.7 of standard deviation, which is almost three times as many as ASPM algorithm. So we can see that the ASPM algorithm can get lower standard deviation than the AMP algorithm, which proved that ASPM can obtain better fairness to concurrent streams with diverse access rate.

4 Relate Works

4.1 History-Based Prefetching

History-based prefetching predicts future access by learning history past accesses. Recording the history access requires a significant amount of memory. Besides, the performance of prefetching is dependent of the predictive accuracy. The low predictive accuracy always degrade performance. These techniques always use extra cache to track addresses [7, 8]. And utilize this information to detect the sequential stream and make prefetching decision. Table-based Prefetching extend then history-base prefetching. Unlikely the above prefetching, Table-based Prefetching algorithms only record the prior I/O request address in table and detect the sequential access pattern by the table, like Tap [13] and STEP [14].

4.2 Hint-Based Prefetching

Hint-based prefetching always exploit the hint provided or captured form the application to learn the spatial locality of the future accesses. Gokul Soundararajan et al. propose the QuickMine [9]. QuickMine capture application contexts and leverage them for context-aware prediction and improved prefetching effectiveness in the storage cache. Patterson et al. propose the TIP2 [15] It uses application disclosed access patterns (hints) to expose and exploit I/O parallelism, and applies cost-benefit analysis to dynamically allocate buffers among three competing demands: prefetching hinted blocks, caching hinted blocks, and un-hinted accesses blocks.

4.3 Adaptive Sequential Prefetching

Zhe Zhang [9] found that there exist strong similar between supply chain management (SCM) and multiple streams prefetch. It applies SCM principles design two prefetching method, Equal Time Supplies (ETS) and Equal Safety Factors (ESF). The former allocate prefetching resources proportional to the access rates and the later allocate prefetching resources proportional to the standard deviations of their requests. SARC [7] and AMP [8] both focused on balancing cache allocation between random and sequential accesses and among multiple sequential streams. SARC design a self-tuning, simple to implement and locally adaptive prefetching method, named SARC. It dynamically and adaptively partitions the cache space amongst sequential and random streams so as to reduce the read misses. AMP analysis the best prefetching degree and trigger distance when it's given streams access rate and cache size. Also it accordingly design and implement an adaptive prefetching algorithms named AMP.

5 Conclusions

Sequential prefetching is widely used in modern storage systems. In this paper, we argued the un-fairness and performance degradation introduce by the concurrent streams with diverse access rate. We also present a novel, adaptive algorithm, named ASPM. We have demonstrated the performance and fairness under a series of wide ranging experiments. The experiment result show that our algorithms can both improved the performance and fairness for the concurrent streams with diverse access rate.

Acknowledgements. We would like to thank the anonymous reviewers for their helpful comments in reviewing this paper.This work was supported by National Science Fund for Distinguished Young Scholars No.61025008, the National Basic Research Program of China (973 Program) under Grant No.2011CB302301, The National Key Technology R&D Program No.2011BAH04B02, Electronic Development fund of Information Industry Ministry.

References

1. Carr, R.W., Hennessy, J.L.: WSCLOCK-a simple and effective algorithm for virtual memory management. SIGOPS Oper. Syst. Rev. 15, 87–95 (1981)
2. O'Neil, E.J., O'Neil, P.E., Weikum, G.: The LRU-K page replacement algorithm for database disk buffering. SIGMOD Rec. 22, 297–306 (1993)
3. O'Neil, E.J., O'Neil, P.E., Weikum, G.: An optimality proof of the LRU-K page replacement algorithm. J. ACM 46, 92–112 (1999)
4. Jiang, S., Zhang, X.: LIRS: an efficient low inter-reference recency set replacement policy to improve buffer cache performance. SIGMETRICS Perform. Eval. Rev. 30, 31–42 (2002)
5. Zhou, Y., Philbin, J., Li, K.: The Multi-Queue Replacement Algorithm for Second Level Buffer Caches. In: Proceedings of the General Track: 2002 USENIX Annual Technical Conference, pp. 91–104. USENIX Association (2001)
6. Li, Z., Chen, Z., Srinivasan, S.M., Zhou, Y.: C-Miner: mining block correlations in storage systems. In: Proceedings of the 3rd USENIX Conference on File and Storage Technologies, p. 13. USENIX Association, San Francisco (2004)
7. Gill, B.S., Modha, D.S.: SARC: sequential prefetching in adaptive replacement cache. In: Proceedings of the Annual Conference on USENIX Annual Technical Conference, p. 33. USENIX Association, Anaheim (2005)
8. Gill, B.S., Bathen, L.A.D.: AMP: adaptive multi-stream prefetching in a shared cache. In: Proceedings of the 5th USENIX Conference on File and Storage Technologies, p. 26. USENIX Association, San Jose (2007)
9. Zhang, Z., Kulkarni, A., Ma, X., Zhou, Y.: Memory resource allocation for file system prefetching: from a supply chain management perspective. In: Proceedings of the 4th ACM European Conference on Computer Systems, pp. 75–88. ACM, Nuremberg (2009)
10. Bucy, J.S., Schindler, J., Schlosser, S.W., Ganger, G.R., Contributors: The DiskSim simulation environment version 4.0 reference manual (2008)
11. Lab, O.S.D.: Iometer I/O performance analysis tool (2003)
12. Axboe, J., Brunelle, A.D., Scott, N.: blktrace(8) - linux man page (2013)
13. Li, M., Varki, E., Bhatia, S., Merchant, A.: TaP: table-based prefetching for storage caches. In: Proceedings of the 6th USENIX Conference on File and Storage Technologies, pp. 1–16. USENIX Association, San Jose (2008)
14. Liang, S., Jiang, S., Zhang, X.: STEP: Sequentiality and Thrashing Detection Based Prefetching to Improve Performance of Networked Storage Servers. In: Proceedings of the 27th International Conference on Distributed Computing Systems, p. 64. IEEE Computer Society (2007)
15. Patterson, R.H., Gibson, G.A., Ginting, E., Stodolsky, D., Zelenka, J.: Informed prefetching and caching. SIGOPS Oper. Syst. Rev. 29, 79–95 (1995)
16. Wachs, M., Abd-El-Malek, M., Thereska, E., Ganger, G.R.: Argon: performance insulation for shared storage servers. In: Proceedings of the 5th USENIX Conference on File and Storage Technologies, p. 5. USENIX Association, San Jose (2007)

Incremental Truth Discovery for Information from Multiple Data Sources

Li Jia, Hongzhi Wang, Jianzhong Li, and Hong Gao

Harbin Institute of Technology
0jialijiali@163.com, {wangzh,lijzh,honggao}@hit.edu.cn

Abstract. In practice, input data may come incrementally during data integration, static algorithm can't adapt for this situation. So, to make truth discovery algorithm more practical, we present an incremental strategy in multisource integration using boosting like ensemble classifier. Our algorithm is adaptive for different update situations by considering concept drift in learning process. Our based model can treat entities inconsistently for a source also. These make truth finding more effective without repetitive computation.

Keywords: Truth finding, concept drift, data integration, incremental algorithm.

1 Introduction

In the process of integrating data from different sources, for one entity, some of its description may be conflicting. Thus we need to make conflict resolution. In these sources, partial data contains errors or with low quality. Therefore, we have to deal with the problem how to find the truth from conflicting information about the same entity. In practice, we cannot accomplish the whole task at one stroke when data does not completely come. For example, in data stream, we need a dynamic computing algorithm, since data sources may be integrated one by one, and the data of all sources may come batch-wise. Such circumstance requires incremental methods. Hence this paper proposes an incremental algorithm to solve the *truth finding problem*. The methods presented before can be used as unsupervised machine learning and static learning in a majority. But their static strategy cannot adapt for incremental situation.

We combine truth discovery with machine learning strategy (boosting multiple weak classifier into a strong one)[1] to train model. In reality, we can treat the *truth finding model* as a binary classifier which can infer the truth *true* or *false*. In consequence, it can generate a boosting-like ensemble classifier. In each sub-classifier we train a base model in which one source can invest different weight into its objects. Our model is based on Bayesian Inference and considers both accuracy and efficiency. It not only is suitable for different *update types* (we define three *update types* as

[1] Our weakly classifier may be not with low accuracy but be a sub-classifier in a specific dataset.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 56–66, 2013.

data stream, big task updating and sources updating according the granularity of updating), but also solves a problem, *concept drift*, which is a challenge in incremental machine learning.

2 Basic Framework

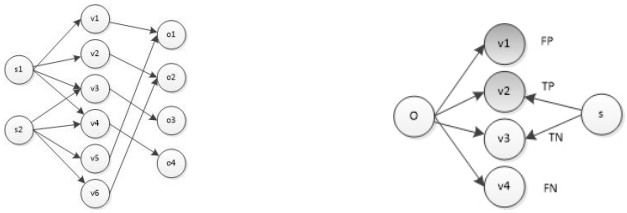
2.1 Definitions

In this section, we formally define the problem to be solved as well as related concepts. In general, a data source is the information provider of some *data objects*. First we give definition of the input data of our algorithm and the relationship among three important parties, *sources*, *objects* and *values* (We can see this from Fig.1). All of these three components know each other in advance. Our input data comes from different data sources, each of the sources is a set of *claims*. The relationship is shown in Definition 1.

Definition 1. Let input data $D = \{source_1, source_2, ...source_m\}$, and each *source* be a set of *claims*. $source_i = \{claim_1, claim_2,..., claim_n\}$, and each *claim* can be a *record* $r = \{source\_id, object\_id, content\}$.

A value refers to the value of a property of a real-world entity. Each *claim* is an instance of a *value*. In different *sources*, a *value* may have different *claims*. In common $m<<n$.

We illustrate the relationship among *sources*, *objects* and *values* in Fig. 1. Fig. 1 (a) describes the inclusion relationship and many-to-many relationship among the three variables. Fig. 1 (b) describes the two-side quality of sources. False values are marked shadow.



(a) many-to-many relationship (b) the two-side quality of sources

Fig. 1. The relationship among *sources*, *values*, and *objects*

In our method, we treat *truth* as a random variable and infer the probability of it to make predictions.

Definition 2. The *truth label t* is a Boolean random variable about how true a *value* is which can be computed by the Bayesian model. *Observation o* is also a Boolean random variable. It reflects true possibility of a specific *claim* when a truth is certain and *t* is its label.

Definition 3. By comparing between *o* and *t,* we can define four attributes *True Positives* (*TPs*) if $o = True$ and $t = true$, *False Positives* (*FPs*) if $o = false$ and $t = true$. In the same way, we define *False Negatives* (*FNs*) and *True Negatives* (*TNs*).

The counts of these four attributes inherently express the *quality* of each source. Intuitively, better sources can provide more *true claims* with high *TPs* and high *TNs.*

For the simplicity of discussion, we give some assumptions which are reasonable. First, the quality of each object in every data source is known. Second, each source provides only one claim of an object, and the entity resolution has been processed before truth discovery. Then, if the data quality is too low, we cannot get high-quality results from it. Therefore, the obtained data must be reliable enough for prediction. It assures that the knowledge discovered from data is useful and suitable for analysis.

2.2 Bayesian Model

In this section, we use a Bayesian Graphic Model to infer the truth probability of a given value according to the quality of sources. We use Bayesian Model to update sources' *quality* by prior learned knowledge. We define our proposed model for *truth finding* and *source quality computation*. We first give the evaluation strategy of *sources* and then propose the algorithm how to infer *truth* by iteratively computing source *quality* which inherently contains comparing *observation* and *truth label*.

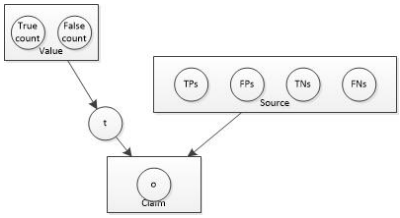


Fig. 2. The probabilistic graphical model

The basic idea is illustrated in Fig. 2, which shows the inference process. This process has four steps. First, we compute the *quality* of *values*. Then the truth for each value is selected and labeled. Second, we compute the *quality* of *sources*, which is represented as the two-sided *counts* defined in Definition 3. Third, we compare the *observation* with existing *truth* then modify the *quality* of *values* and *sources*. In the last step, if the probability of *truth* is above a given *threshold*, we treat it as the *truth*. The truth finding requires the computation of $p(t = true|o, s)$, which is the probability of *t* being true. $p(t = true|o, s)$ changes with the comparison between *observation* and current *truth* . In this round, *truth* is selected and the probability of it is computed accordingly.

To compute the probability $p(t = true|o, s)$, we can compute p_t instead. Let the probability of the current *truth t* is *true* be equivalent to p_t:

$$p_t \propto p(t = true|o, s) \tag{1}$$

$p_{\bar{t}}$ is defined in the same way as follows.

$$p_{\bar{t}} \propto p(t = false|o, s) \tag{2}$$

We compute p_t with the *quality* of the *sources*. Then for each *observation o* of claim the p_t in current iteration round is p_t^{now} and p_t^{now} is computed as follows.

$$p_t^{now} = \frac{p_t^{before} \times (TPs_c + TPs)}{TPs_c + TNs_c + TPs + TNs} \quad if\ o = true \tag{3}$$

$$p_t^{now} = \frac{p_t^{before} \times (FPs_c + FPs)}{FPs_c + FNs_c + FPs + FNs} \quad if\ o = false \tag{4}$$

Algorithm 1. BaseTF (fundamental inference algorithm)
Input: {*S, O*} or *D* (the input dataset about sources *S* and objects *O*)
Output: The true value for each object in *O*.

```
1 Program BaseTF
2    for each value:
3      Initialize p_t with prior true number;
4      Initialize p_t̄ with prior false number;
5    while oscillation of Result Set is below  :
6       for each source:
7         for each value:
8           for each claim:
9             Compute p_t^now and p_t̄^now;
10          if p_t^now ÷ p_t̄^now ≥ Θ:
11            t is true;
12          else
13            t is false;
14            Update TPs,TNs,FPs,FNs;
15        Update the ResultSet
16     for each value:
17       if t is true: p(t = true) = (p_t − p_t̄) / p_t;
```

The complexity of this algorithm is O(*m*n*), where *m*n* is the number of claims of all sources.

3 Incremental Algorithms (INC)

3.1 Preparation For Increment

Incremental truth discovery requires that the *based classifier* should not be retained when new data comes. How to achieve this goal? We cannot only use existing model to predict truth or train the model in new data simply due to its *accuracy*, *efficiency*, and *data drifting*. Then, we propose a multi-classifier voting strategy to solve this problem.

In this algorithm, we use multiple classifiers. For each classifier M_i, if $p(t = true) > \tau$, we take M_i for availability at particular object. Otherwise, M_i does not involve voting. Similarly, if $p(t = true) > \theta$, we consider that M_i has high weight in voting for this object. Thus, we can divide the output of M_i into three parts (*part A* as very sure, *part B* as classable, *part C* as uncertain), each *truth* labeled *A*, *B*, *C* with its corresponding part.

We denote $W_s(o)$ as the *weight* about the source's votes on its object. If the *truth probability* of the values in the object of source is high, the source's votes on this object will also be high.

$$W_s(o) = \log_2 |V_s| \frac{\sum_v p(V_s(o) = true)}{|V_s|} \tag{5}$$

Also, the hardness $H(o)$ denotes how hard the truth of an object can be inferred from the sub-model. The harder a classifier can infer the truth on *o*, the less *weight* it has in voting.

$$H(o) = \rho \times \frac{|W_s|}{\log_2 \sup(v \to o) \sum_s W_s(o)} \tag{6}$$

Where $\sup(v \to o)$ is the support in existing data about an object and values. ρ is an adjustment parameter.

3.2 Voting Algorithm

Without any knowledge about the truth, we cannot evaluate the accuracy by the comparison with labeled dataset. Thus we evaluate the accuracy using data mining techniques. Our basic idea is voting, using each sub-classifier trained with different data votes for the final truth. Our algorithm is divided into three steps, *initialization*, *voting* and *update*. The basic idea is that, each sub-classifier only votes for objects that it contains. The *weight* of each sub-classifier depends on its *error-rate* and the *hardness* of an object specific sub-classifier can infer its *truth*. Certainly $H(o)$ is inherently including *weight* of sources (as shown in (5) and (6)). The general flow chart below is shown in Fig.3. When a new input comes, the three modules of our algorithm are executed in this order. Their intermediate results are stored in the *Result Set*.

In the remaining part of this section, we will discuss the details of these three modules.

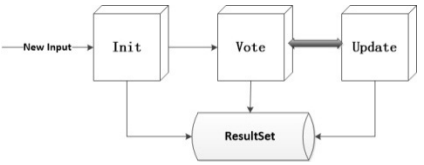


Fig. 3. The relationship among each module of Voting Algorithm

3.2.1 Initialization

In this stage, we train sub-classifier with existing data using the Bayesian Model ntioned above. And giving label *A*, *B*, *C* for each *truth* (the labels is defined in 3.1).

Algorithm 2. Initial
Input: The new input data with corresponding classifier M_i
Output: The true value for each object in M_i
The *label* for each *truth*

```
1  program initialization(existing data)
2     while truth is still vibration:
3     {source quality don't change in the medium process}
4       Compute the counts of quality of sources;
5       Compute the truth of each value;
6
7     {Divide the output into three parts}
8     for each object:
9       Assign o whose p(t=true)>0.5 with an label;
10      Compute H(o);
11    {keep the attribute of O in the underground area}
```

3.2.2 Voting

In this stage, we discover the new *truth* according to new data object. We first compare the *labels* among each classifier. Each classifier only votes for the objects that others do not has a higher *label*. For example, if *label* of M_i on *o* is B and M_j's *label* is C, then M_j will not take part in voting for *o*. Then we compute the *confidence* of each value and choose the one with maximal value as the *truth*. But if the *probability* of the given *truth* is low, we still keep *unknown state* on it.

Algorithm 3. Voting
Input: $M_1, M_2, \cdots, M_i$
Output: Update *true value* for each object with highest *confidence*

```
1 Program vote(
2   Classifier M_1, M_2, ..., M_i, classifier M_new, newInput)
3   Compare the label among the classifiers;
4   Got the classifiers with max label;
5   for each object:
6     Compute H_{M_i}(o) and E(M_i) //based on formula (6) and (7)
7     for each value:
8       for each classifier:
9         if the classifier's truth is this value
10            confidence(v)+= H_{M_i}(o)E(M_i);
11      Compare the confidence among values;
12      Give the truth v with max confidence;
13      {the same to before in section 2}
14      Compute the probability of v;
15     if p(v) < Ψ, give up v
```

In this algorithm, we have trained some classifier on existing data.

3.2.3 Updating

In this stage, we compute the *error rate* for each classifier to evaluate the *quality* of classifier. Without any testing samples, the classifier's *quality* can be assessed by other classifiers and its own inner consistency. We can compute $E(M_i)$ by 1-*accuracy* of M_i .

$$E(M_i) = 1 - \frac{\sum TNs + TPs}{\sum TNs + TPs + FNs + FPs} \tag{7}$$

So we can give the *truth* by iteratively run *voting* and *update* in the end.

4 Concept Drift Handling

Concept drift will affect truth discovery. In this section, we propose the strategy of concept drift handling. At first, we use an example to illustrate concept drift. The concept may drift with time and its probability of occurrence is high in some specific circumstance. The data quality can be derived from data. Thus we can properly draw

conclusion that the data can say whether it can be trained or whether its training classifier is drifting. Now, we classify the data into four classes, *sufficient and no drift, sufficient and drift, insufficient and no drift, insufficient and drift.*

To judge whether a data block is drifting, we define the concept of *Drift* and *Sufficiency*. Since *sufficiency* is hard to decide directly from data, we usually judge it by experience.

$$Suf(data) = E(M_{data}) \times \log_2 data_{size} \tag{8}$$

For defining *Drift of* current *data*, we can use votes of existing classifiers, others' data also can say about the data's *Drift* in a specific data block.

$$Drif(data) = \sum_{M_i} Drif(M_i) \times dis(M_i, data) \tag{9}$$

$Drif(M_i)$ can be computed by its *Drift* of its previous trained classifier. And the *Distance* between M_i and *data* can be computed as follows.

$$dis(M_i, data) = \sum_{intersect} \{+1(sametruth) | -1(diftruth)\} + \log_2 \sum_{disjoint} 1 \tag{10}$$

In this formula *intersect* delegate the set of *intersecting objects* and *disjoint* delegate the set of $disjoint$ *objects*. We divide the object set (union set of the data of M_i and the current *data*) into two parts. One is of *intersecting objects* and one is of *disjoint* objects. In *intersect* dis add one if the same truth exists, otherwise minus 1. $Disjoint\ set$ is settled likewise.

If the new input is *Sufficient and No drift*, we can apply the old classifier for prediction without risk. In the second situation, we should train a new model from new data. In the third case, there are two possibilities. The first one is that old model is trained from sufficient data. Thus the old model is kept. For the second possibility that new data and old data are mixed, new model is to be trained. In the last situation, we can give an "Un-biased" Selection Framework with model evaluation. Based on this discussion, we describe the incremental algorithm for truth finding with consideration of concept drifting. We also use the framework of INC. Based on the updating form, we classify updating into three cases. The incremental truth discovery algorithm with consideration of concept drift is denoted by INC- Drift 1 to INC- Drift 3. They are discussed as follows.

(1) When a new source is integrated in, it corresponds to the *sufficient and drifting* case. We can train *FN* from new data. Train *FN+* from new data and selected consistent old data. Assuming *FO* is the previous most accurate model; *FO* is updated with the new data. The updated one is denoted by *FO+*. The final classifier is chosen among the four candidate models {*FN*, *FN+*, *FO*, and *FO+*} with lowest *error-rate*.

(2) Some information of old sources has not arrived. Our incremental algorithm is suitable for this case. The voting strategy can be applied directly.

(3) In the procession of one task, if previous data is sufficient and the input is like a data stream, we can accelerate the algorithm by ignoring middle process. We can make decision directly when a new tuple comes for the *sufficient and no drift* case.

5 Experiments

To test the performance algorithm, we run extensive experiments. We run our method on a PC with Intel(R) Core(TM) i5-2400 cpu 3.10GHz and 4GB memory. And our operation OS is windows 7. All code is realized by C++.

Our incremental algorithm can be divided into four categories. INC denotes the basic Incremental classification algorithm without *concept drifts* handling. INC- Drift 1 to INC- Drift 3 denotes the algorithm with concept drift handling the cases 1-3 in Section 4.

We test our algorithms in two real datasets. In each dataset, we perform tests on the comparison of *accuracy* among methods, parameter adjustment and efficiency. We run our algorithm in different environment. We use both multi-truth set, Movie Director Dataset, and single truth set, Book Author Dataset. We also change thresholds from 0.5 to 1.0.

- Experimental Results on Movie Director Dataset

This data set is collected by using the Bing movies vertical for surfacing reviews, meta-data and entity actions such as “rent” and “stream”.

Table 1. The Accuracy of each algorithm and the Threshold is 0.5

Algorithm	Accuracy %	Situation
Truth Finder	77	All data is in preparation
Ivotes	86	
3-Estimates	85	
BaseTF	89	
INC	82	Sources added one by one
INC	75	Big task coming one by one
INC	71	Data stream
INC-Drift 1	86	Sources added one by one
INC-Drift 2	87	Big task coming one by one
INC-Drift 2	79	Data stream

We can see that in traditional situation without any increment, algorithm goes with a litter higher *accuracy*. All of Truth Finder[2], Ivotes, 3-Estimates[9], BaseTF perform the best. And BaseTF is relatively better because of multi-truth dataset and some latent advantages.

Basic Incremental algorithm can behave well in some cases but terrible unexpected sometimes. Therefore, *concept drift handling* is in demand. Accuracies of INC-Drift 1 and INC-Drift 2 rise significantly comparing with INC. The results of our experiment show that our increment method is applicable in practice.

- Experimental Results on Book Author Dataset

Table 2. The Accuracy of each algorithm and the Threshold is 0.5

Algorithm	Accuracy %	Situation
Truth Finder	92	All data is in preparation
Ivotes	89	
3-Estimates	88	
FIA	95	
INC	90	Sources added one by one
INC-Drift 1	92	Sources added one by one
INC-Drift 2	94	Big task coming one by one
INC-Drift 3	85	Data stream

- Efficiency Test

We test run time of both two datasets. The experimental results are shown in Figure 6.

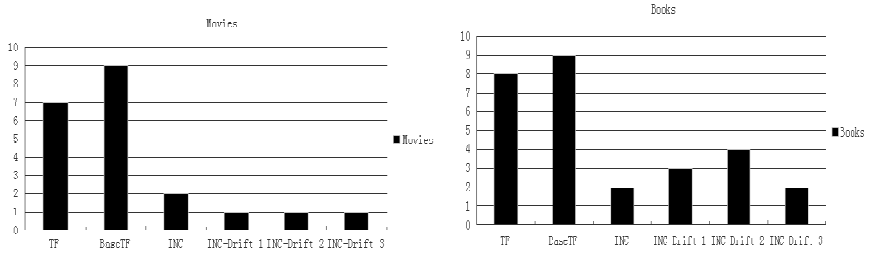


Fig. 4. Performance Summary

6 Conclusions

Truth discovery is a necessary stage in big data integration from multi-sources. Incremental is strongly needed in practical use but few authors consider this problem.

In our paper, we propose an incremental algorithm which make truth discovery more effective with little accuracy loss. From our work, we can see that our incremental algorithm can be adaptive to diverse of situations. It does not only can be applied using Bayesian inference model, but also can work in tradition truth-find methods.

Acknowledgement. This paper was partially supported by NGFR 973 grant 2012CB316200, NSFC grant 61003046, 61111130189 and NGFR 863 grant 2012AA011004. Doctoral Fund of Ministry of Education of China (No.20102302120054).

References

1. De Corte, E., Op't Eynde, P., Verschaffel, L.: Knowing what to believe (when you already know something). In: COLING 2010 Proceedings of the 23rd International Conference on Computational Linguistics, pp. 877–885. Association for Computational Linguistics, Stroudsburg (2010)
2. Yin, X., Han, J., Yu, P.S.: Truth discovery with multiple conflicting information providers on the Web. In: Proc. of SIGKDD (2007)

3. Dong, X.L., Berti-Equille, L., Srivastava, D.: Integrating conflicting data: the role of source dependence. PVLDB, 2(1-2) (2009)
4. Zhao, B., Rubinstein, B.I.P., Gemmell, J., Han, J.: A Bayesian approach to discovering truth from conflicting sources for data integration. J. Proceedings of the VLDB Endowment Hompage Archive 5(6), 550–561 (2012)
5. Balakrishnan, R., Kambhampati, S.: SourceRank relevance and trust assessment for deep web sources based on inter-source agreement. In: WWW 2011 Proceedings of the 20th International Conference on World wide Web, pp. 227–236 (2011)
6. Wu, J., Ding, D., Hua, X.S., Zhang, B.: Tracking concept drifting with an online-optimized incremental learning framework. In: SIGMM, pp. 33–40 (2005)
7. Scholz, M., Klinkenberg, R.: Boosting Classifiers for Drifting Concepts. J. Intelligent Data Analysis 11, 3–28 (2007)
8. Galland, A., Abiteboul, S., Marian, A., Senellart, P.: Corroborating information from disagreeing views. In: WSDM, pp. 131–140 (2010)
9. Galland, A., Abiteboul, S., Marian, A., Senellart, P.: Corroborating information from disagreeing views. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 131–140. ACM (2010)

Continuous, Online Anomaly Region Detection and Tracking in Networks*

Shuiyuan Xie[1,2], Xiuli Ma[1,2,**], and Shiwei Tang[1,2]

[1] Key Laboratory of Machine Perception(Peking University), Ministry of Education
[2] School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, China
xsy2510@gmail.com, maxl@cis.pku.edu.cn

Abstract. In many real networks, the detection and tracking of unusual phenomena, such as the diffusion of contamination and the spreading of disease, is one of the key feature users are great interested in, which is called anomaly with technical terms. In this paper, we present a framework to detect and track anomaly region continuously. First, we build a state transition graph to summarize network's operating regularity, that is, network stays in a state for a period of time and alternates among states over and over again, which exists in many real networks. Second, we employ the state transition graph to predict network's next state. While comparing expected state and current state, we present suspicious region and its anomaly probability. We evaluate our approach on a real water distribution network from the Battle of the Water Sensor Network (BWSN). Experiments show that our approach is effective, efficient and scalable to detect and track anomaly region.

Keywords: Anomaly region detection and tracking, state transition graph.

1 Introduction

In real world, there are various monitoring networks to monitor environment, traffic, disease and so on. One of the goals administrators are often interested in is the detection and tracking of unusual phenomena, that is, detecting a geographically abnormal region is of great interest to users, which is called anomaly region with technical terms.

Several studies have been done in detecting anomaly region. Some focused on finding outliers separately and then getting anomaly region based on topology [1][2][3]. Some predefine event's spatio-temporal pattern and employ pattern matching to detect event [4]. In this work, we focus on detecting anomaly region by the diffusion degree of region. To the best of our knowledge, the discovery of suspicious region and the detection of anomaly region based on diffusion degree have not been investigated before.

* This work is supported by the National Natural Science Foundation of China under Grant No.61103025.

** Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 67–78, 2013.

Our framework to detect anomaly relies on two important criterions. First, many networks run regularly. Network has some internal states, stays in a state for a period of time and alternates among these states over and over again [5]. Many networks fall into this category. So we can unveil network's internal states and build a state transition graph to summarize network's operating regularity. As Figure 1 illustrates, we mine a real water distribution network from the Battle of the Water Sensor Network (BWSN) and learn that the network has six hidden internal states and alternates among these six states. Second, when an anomaly happens, it influences its neighbors firstly and diffuses to further place as time goes by. We illustrate this by an example in Figure 2. By adding a contamination event at node 31 on EPANET [6] which is a tool to simulate how water distribution network runs, we find contamination diffuses downstream to the region {32, 33, 58, 59, 60, 61, 67, 68, 91, 92}. So we focus on detecting the region where sensory data doesn't change as expected and diffuses as time goes by. With a time series of states, we use Markov models to predict network's next state and find the suspiciously diffused region by comparing predicted state and current pattern continuously. What's more, we give the suspiciously diffused region an anomaly probability based on diffusion degree.

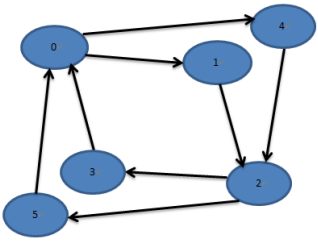


Fig. 1. state transition graph

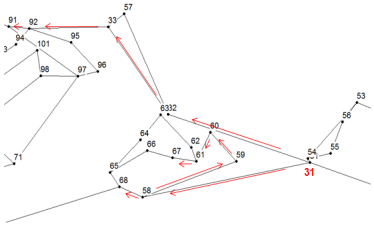


Fig. 2. diffused anomaly region

In summary, the contributions we make in this paper are the following:

1. We reveal network's hidden internal states and build a state transition graph to summarize how network runs in the latest sliding window.
2. We use Markov models to predict network's next state based on state transition graph.
3. We detect and track network's anomaly region with the idea of diffusion.
4. We conduct extensive experiments to verify the effectiveness, efficiency and scalability of the proposed approach.

The rest of the paper is organized as the follows. Section 2 discusses the related work. Section 3 introduces some necessary definitions and formalizes the problem. Section 4 presents how our algorithm summarizes the network's sensory data continuously, learns a state transition graph and uses the model to detect and track anomaly region. Section 5 shows the experimental evaluation and section 6 concludes.

2 Related Work

Anomaly detection is an important problem that has been researched within diverse research areas and application domains. In networks, some work has been done. Sakaki at al. [1] proposed an algorithm to detect earthquake and typhoon real-time with microblogging data. It considers each Twitter user as a sensor and chooses some related sensors each time to estimate an event probability and the center of the event. Liu at al. [2] proposed an algorithm to detect outliers based on spatial and temporal properties and then construct outlier trees to uncover causal relationships among these outliers. Franke at al. [3] presented a system that gives every sensor an outlier degree and constructs outlier regions with outlier degree threshold and topology. They all first find the outlier set and then construct anomaly region based on spatial relationship, which needs massive computation to judge every node is an outlier or not. Xue at al. [4] predefined the event's spatio-temporal pattern and used pattern matching method to detect event, which is not appropriate in some networks since it may be hard to predefine anomaly pattern in these networks.

3 Problem Definition

In this section, we formally introduce some necessary notations and formulate the problems.

First of all, we assume the network infrastructure is as following:

In a network *SN*, there are *m* nodes $SN=\{1,\ldots,m\}$. Each node $s \in SN$ monitors some variables, such as pH, chlorine, and total organic carbon. In the proposed framework, we assume one-dimensional sensory data, i.e., the chlorine variable is monitored by all nodes. However, the proposed method can by extended to multi-dimensional data as well. All nodes periodically stream their data to a central server for processing and analysis.

Table 1. Description of notation

Symbol	Desription
SN	the set of nodes in network
m	Number of nodes
SP_t	the snapshot pattern at time t
RP_r	The r-th representative pattern
E_r	The r-th segment
S	A state
G	A state transition graph
ε	Anomaly probability threshold

3.1 Notation and Definition

Definition 1. [7]. *(SNAPSHOT PATTERN) A snapshot pattern* SP_t *is a clustering* C *of m nodes at time t, which partitions the m nodes into k disjoint set* $\{c_1, c_2 \ldots c_k\}$, *that is,* $\bigcup_i^k c_i = SN$ *and* $c_i \cap c_j = \varphi$ *for all* $i \neq j$.

We employ the snapshot pattern to depict nodes' spatial distribution structure at every timestamp by clusters of the raw sensory data of m nodes. To denote the relationship of nodes, we build a 0-1 matrix, in which *matrix[i][j]=1* means node i and j are in different clusters, otherwise in the same cluster. Since nodes' correlation changes when anomaly happens, we consider the clusters' structure of snapshot pattern would change as well, which could be of great avail for detecting anomaly.

As sensory data changes smoothly and has strong spatial correlation, the clusters' structure of snapshot changes flatly as time goes by. So we group consecutive timestamps into segments to track how the structure of snapshot patterns evolves over time. Within each segment, we use a new pattern to represent it.

Definition 2. [7]**.** *(REPRESENTATIVE PATTERN) A representative pattern is a typical pattern RP_r that can best represent the snapshot patterns of segment E_r.*

The "typical" and "best" means that the representative pattern is a constructed snapshot pattern which is the most consistent both in structure and data with the snapshot patterns it represents. The pattern structure and data feature of RP_r are the summarization of the snapshot patterns in E_r.

By analyzing representative pattern stream, we find the stream is regular, even periodic, as shown in the prior work [7], which supports our hypothesis that many networks run regularly and vice versa. So we employ a state transition graph to represent this regularity.

Definition 3. *(STATE TRANSITION GRAPH) A state transition graph is a directed graph, in which every vertex is a state at which network stays for a time, and a directed edge from S_1 to S_2 with a weigh w denotes that the network transferred from S_1 to S_2 w times in the latest sliding window.*

Since the snapshot patterns in a segment are so similar that they can be summarized by a representative pattern, we consider the timestamps in a segment are all in a state. So we employ representative pattern's structure to express state for conciseness, that is, representative pattern and state are many-to-one relationship.

3.2 Problem Formulation

Our ultimate goal is to detect and track anomaly region. Thus, we address the problem as follows:

Problem 1. *(ANOMALY REGION DETECTION AND TRACKING).When an anomaly happens in a network, how to detect the anomaly region r as soon as possible and track the diffusion of region r continuously.*

We first mine network's operating regularity, and then predict network's next state, and compare expected state with current pattern continuously to find the suspiciously diffused region finally. The details will be present in the next section.

4 Approach

In this section, we will describe the framework with three parts: (1) state transition graph construction; (2) state prediction; (3) anomaly region detection and tracking. The details are as follows:

4.1 State Transition Graph Construction

To distinguish normal variation and anomaly, we first learn network's normal operating regularity. As mentioned above, many real networks have some internal states, stay in a state for a period of time and alternates among these states over and over again. So we construct a state transition graph to represent this regularity.

As the prior work [7] describes, we cluster sensory data at every timestamp which we call snapshot pattern, and then split the snapshot pattern stream into segments incrementally represented by representative pattern. The algorithm to split snapshot pattern stream into segments incrementally is based on one form of the Minimum Description Length (MDL) principle which estimates the joined storage cost and split storage cost to decide split or not when a new snapshot pattern arrives.

In this paper, we study how to map a segment to the corresponding state. The guiding principle is still the Minimum Description Length (MDL) principle. If an old state can store a segment with less cost than a new state that uses the current representative pattern's structure, then we consider that this segment is in this state. Else we add the new state to state transition graph for storing this segment. The storage cost we adopt is based on information theory. The detail formula of the two choices' cost is as follow:

$$cost(S_{old}, E_r) = \log^* I + \log^* l_r + \sum_{i=t_r}^{t_{r+1}-1} cost(SP_i \ XOR \ S_{old})$$

$$cost(S_{new}, E_r) = \log^* I' + \log^* l_r + cost(RP_{new}.mtr) + \sum_{i=t_r}^{t_{r+1}-1} cost(SP_i \ XOR \ RP_{new})$$

where $\log^* I$ and $\log^* I'$ are the cost to store the indices of states and $\log^* l_r$ is the cost to store the length of segment. The storage cost of a 0-1 matrix X is accurately estimated as $m^2H(X)$ where $H(X)$ is the entropy of the matrix. Additionally, two integers need to be stored: the matrix's size m, and the number of ones in the matrix, denoted as $|X|$. So the storage cost of a 0-1 matrix is:

$$cost(X) = \log^* m + \log^* |X| + m^2 H(X)$$

$$H(X) = -(\rho \log \rho + (1-\rho)\log(1-\rho)), \ \rho = (\sum_{i=1}^{m}\sum_{j=1}^{m} X[i][j]) / m^2$$

Algorithm 1. ConstructGraph (state transition graph G, new coming segment E_r)

1. **Initial**: S_1=RP_1, insert S_1 into G, $S_{pre} = S_1$ //preceding state
2. **Find** the state S_i to store E_r with least cost in G
3. **If** $Cost(S_i,E_r)<Cost(S_{new}=RP_r,E_r)$ **then**
4. $S_{curr}=S_i$ //current state
5. **Else** //add a new state
6. *add* $S_{new}=RP_r$ *into G*
7. $S_{curr}=S_{new}$
8. **End if**
9. **If** not edge from S_{pre} to S_{curr} in G **then**
10. Add an edge from S_{pre} to $S_{curr,}$ and set its degree with 0
11. **End if**
12. The degree at edge(S_{pre}, S_{curr}) ++
13. //maintain a sliding window
14. **If** r>w **then** //w: the transition number in a sliding window
15. Get the state at segment E_{r-w} and E_{r-w+1}, labeled with S' and S''
16. The degree at edge(S', S'')—
17. **End if**
18. $S_{pre}=S_{curr}$

Output: state transition graph G

An important issue in data stream is time-variation, i.e., the data characteristics may drift with time. To handle this phenomena, we adopt a finite-memory state transition graph construction model and incrementally update its parameters so that it reflects the characteristics of the most recent data [8]. The main idea is to maintain the state transition graph using a sliding window of the most recent W transitions and update the parameters of the state transition graph when new segment is available.

The detailed algorithm to construct state transition graph is shown in Algorithm 1. Given a new segment E_r, we first find the most similar state S_i to describe it. If the cost to store E_r with S_i is bigger than the cost of a new state S_{new}=RP_r, then we add S_{new} to the state transition graph (Line 2 to 13). It is also essential to maintain the state transition graph in the latest sliding window (Line 15 to 18).

4.2 State Prediction

Based on state transition graph, we predict the state for a period of time in future, which reduces massive computation at every timestamp for predict next timestamp's value, compared with other methods. We adopt a finite-memory Markov model [8] to predict current state, based on the characteristics of the most recent sensory data.

At time t, the most recent W+1 states are S_{i-W}, S_{i-W+1}, … , S_i, and current representative pattern is RP_r, we predict an appropriate state for a period of time in future. Assume S_{i-W}, S_{i-W+1},…, S_i are generated by a Markov-chain P, it can be shown that the maximum-likelihood estimation (MLE) for P_{ij} is

$$\hat{P}_{ij} = n_{ij} / \sum_k n_{ik}$$

where n_{ij} is the number of observed transitions from state S_i to state S_j among the W transitions. And the most appropriate state is the state S_j:

$$j = \arg\max_{j} \hat{P}_{ij} * P(RP_r \mid S_j)$$

where $P(RP_r|S_j)$ is the probability of observing RP_r when the network is in state S_j. We adopt the similarity degree to calculate $P(RP_r|S_j)$:

$$P(RP_r \mid S_j) = 1 - (\sum_{p=0}^{m-1}\sum_{q=0}^{m-1}(RP_r \ XOR \ S_j)[p][q]) / m^2$$

The detailed algorithm to predict current state is shown in Algorithm 2. The other states that have not edge from the preceding sate S_i are pruned since n_{ij} equals zero and so is P_{ij}.

Algorithm 2. PredictState (state transition graph *G,* preceding state S_i, current representative pattern RP_r)

```
1.  predictState = null, maxProb = 0
2.  For state Sj in G
3.      If has edge(Si,Sj) then
4.          p <- Pij*P(RPr|Sj)
5.          If p > maxProb then
6.              predictState = Sj
7.              maxProb = p
8.          End if
9.      End if
```

Output: an predicted state predictState

4.3 Anomaly Region Detection and Tracking

In this section, we describe how to detect and track anomaly region based on predicted state and current representative pattern. The idea is that the affected region diffuses as time goes by when anomaly happens. So we first finds the difference node set between predicted state and current representative pattern at every time, and then finds the diffused region with a probability to indicate its anomaly degree.

Find the Difference Node Set. In the first steps, we compare predicted state and current representative pattern to find the difference node set. For example, {(1,2), (3,4,5)}, {(1,2,3), (4,5)} are two different clusterings, we try to find the difference node set {3} whose node jumps from (3,4,5) to (1,2). Algorithm 3 describes how it works. First, we do XOR operation between their matrices (Line 3) and count every node's pairs that don't equal 0 (Line 4 to 9), which implies the pair's relationship changes between predicted state and current representative pattern. Second, we find the node *maxId* with maximum count (Line 10) which implies its relationship with other nodes changes most intensely and add it to the difference node set. To eliminate

its influence to other nodes, we assign the pairs *(maxId,i)* and *(i,maxId)* which equal 1 to 0 in *diffMtr* and subtract their difference count (Line 13 to 18). This operation continues until there isn't node whose relationship with the remains changes.

Algorithm 3. GetDiffPointSet (predicted state *S*, current representative pattern RP_r)

```
1.  diffSet = { }, diffNums = new int[m], diffMtr = S.matrix XOR RP_r.matrix
2.  for i = 0 to m-1
3.      for j = i + 1 to m-1
4.          If diffMtr[i][j] != 0 then
5.              diffNums[i]++,   diffNums[j]++
6.          End if
7.  maxId <- get the id of maximum value in diffNums
8.  While diffNums[maxId] > 0
9.      diffSet.add(maxId)
10.     For i = 0 to m-1
11.         If diffMtr[maxId][i] != 0 then
12.             diffMtr[maxId][i] = diffMtr[i][maxId] = 0
13.             diffNums[maxId]--,   diffNums[i]--
14.         End if
15.     maxId <- get the id of maximum value in diffNums
```

Output: difference point set diffSet

In the second steps, we partition the difference node set into difference regions based on the topology. So every region is a geographically connected region.

Find the Diffused Region. When a difference region appears, we index its initial state and estimate its diffusion degree. Assume node i is in the difference region, its contribution to diffusion degree is

$$score(i) = (\sum_{p=1}^{q} d_{ip}) / (\sum_{j=1}^{k} d_{ij})$$

where d_{ij} is the distance from node i to j, k is the number of i's neighbors and q is the number of i's neighbors that are in this difference region. The diffusion degree of the region is the sum of all nodes' score in this region:

$$score = \sum_{i} score(i) \cdot$$

Lastly, we evaluate its anomaly degree with a probability by

$$score(i) = 1 - score_{init} / score_{curr}$$

where $score_{init}$ is the diffusion degree of its initial state and $score_{curr}$ is its current diffusion degree. When p is bigger than a given threshold ε, we give a warning.

The detailed algorithm is shown in Algorithm 4.

Algorithm 4. DetectAnomaly (predicted state *S*, current representative pattern RP_r)

```
1.  diffPoints = compare S and RP_r and find their difference point set
2.  diffRegions = partition diffPoints by the topology
3.  for every region in diffRegions //initRegions is a parameter that saves every region's initial
    state
4.      initRegion <- Index its initial state in initRegions
5.      If initRegion == null then
6.          Add region into initRegions
7.          Continue
8.      Else
9.          Calculate its anomaly probability p
10.         If p > ε then
11.             Give a warning and show this anomaly region
```

Output: an anomaly region with a probability p

5 Experiments

In this section, we evaluate the proposed methods on a real water distribution network. Section 5.1 introduces the datasets and Section 5.2 studies the effectiveness, efficiency and scalability of our methods. Every experiment was implemented in JAVA, and was tested on a PC with Intel i5 3.10GHz CPU and 4GB RAM.

5.1 Datasets

The water distribution network (Figure 3(a)) we used is a real water distribution system from BWSN [9] with 129 nodes. We simulate 354 contamination events in the network. First, we simulate the normal distribution of chlorine concentration at all nodes during 20 days with 5760 timestamps by EPANET 2.0 which is developed to accurately simulate the hydraulic and chemical phenomena within drinking water distribution networks. Then, we select a node randomly and inject contaminant at a random timestamp *t*. By simulating the process of contaminant diffusion with EPANET, we get the contaminant concentration at every node. Last, we simulate every node's chlorine concentration in an event as the formula:

$$Z_e(t) = Z_0(t) - Z_c(t) * k * \sigma$$

Where $Z_e(t)$ is the chlorine concentration in the event, $Z_0(t)$ is the originally normal chlorine concentration, $Z_c(t)$ is contaminant concentration, σ is the deviation of chlorine concentration and *k* is an event strength factor which is set as 1.5.

5.2 Evaluations

(1) Effectiveness

We first discuss the effectiveness of our methods by an event happens at node 31 at 5604 timestamp. As Figure 2 and Table 2 illustrate, a suspiciously diffused region {31,58,68} is detected and a warning is given at timestamp 5669 if anomaly threshold ε is set as 0.65. The anomaly region is the downstream region of node 31, which is

accord with general knowledge. In general, once an anomaly region's sensory data changes significantly, the representative pattern's clustering structure changes later. Comparing with expected state, we can find the anomaly region effectively.

Table 2. suspiciously diffused region

timestamp	Suspiciously diffused region	Anomaly probability	Delay time(h)
5636	31,58	0	2.67
5654	31,58,68	0.53	4.17
5669	31,32,57,58,68	0.78	5.42
5684	31,32,57,58,59,68	0.83	6.67
5729	31,32,33,58,59,60,68,92,94,95	0.91	10.42
5731	31,32,33,58,59,60,61,67,68,91,92	0.92	10.58

(2) Efficiency

We compare recall rate and average delay time of our methods with BWSN in this section. All the comparisons are based on the 354 contamination events. BWSN gives a best solution to place sensors in the network. At each sensor selected by BWSN, we adopt CANARY [10] to detect event, which is a system developed by The U.S. Environmental Protection Agency (EPA) and the Sandia National Laboratories. Table 3 shows the comparison between our methods and BWSN. Our methods have a remarkable improvement in recall rate, although our methods spend a little more time to detect events. The curve between event strength factor k and recall rate of our methods and BWSN is shown as Figure 4. Our methods are better than BWSN in all points.

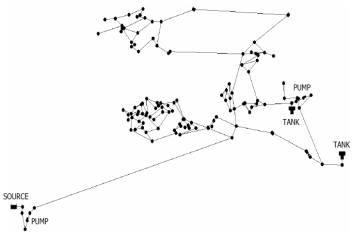


Fig. 3. The distribution water network

Table 3. Comparison

Method	Recall rate	Average delay time(h)
state	0.7966	5.34
bwsn	0.2712	4.13

(3) Scalability

Finally, we illustrate scalability of our methods weighs by CPU time. We do experiments in a network with 1000 nodes picked from Wolf network provided by University of Exeter. As shown in Figure 5, the CPU time curve grows approximately linear when the network size increases. Our methods are scalable to network size.

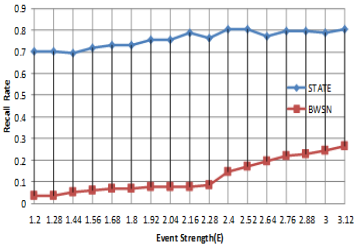


Fig. 4. Recall rate

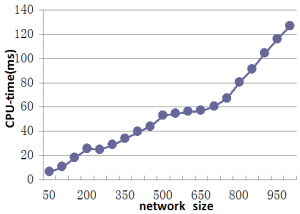


Fig. 5. Network size and CPU time

6 Conclusions

In this paper, we propose a framework to detect and track anomaly region in networks. We mining network's internal states and build a state transition graph. Based on state transition graph, we predict network's next state and compare with current representative pattern for detecting and tracking the suspiciously diffused region. Experiments show our methods are effective, efficient and scalable.

Moreover, our state transition graph is intuitionistic to help users understand how network runs.

References

1. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. In: Proc. of the 19th International Conference on World Wide Web, pp. 851–860 (2010)
2. Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xie, X.: Discovering Spatio-Temporal Causal Interactions in Traffic Data Streams. In: Procedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1010–1018 (2011)
3. Franke, C., Gertz, M.: Outlier Region Detection and Exploration in Sensor Networks. In: Proc. of the 2009 ACM SIGMOD International Conference on Management of Data, pp. 1075–1078 (2009)
4. Xue, W., Luo, Q., Chen, L., Liu, Y.: Contour Map Matching for Event Detection in Sensor Networks. In: Proc. of the 2006 ACM SIGMOD International Conference on Management of Data, pp. 145–156 (2006)
5. Wang, P., Wang, H., Wang, W.: Finding Semantics in Time Series. In: Proc. of the 2011 ACM SIGMOD International Conference on Management of Data, pp. 385–396 (2011)
6. Rossman, L.A.: EPANET2 user's manual: National Risk Management Research Laboratory, U.S. Environmental Protection Agency (2000)
7. Xiao, H., Ma, X., Tang, S., Tian, C.: Continuous Summarization of Co-Evolving Data in Large Water Distribution Network. In: Chen, L., Tang, C., Yang, J., Gao, Y. (eds.) WAIM 2010. LNCS, vol. 6184, pp. 62–73. Springer, Heidelberg (2010)

8. Chi, Y., Yu, P.S., Wang, H., Muntz, R.R.: Loadstar: A Load Shedding Scheme for Classifying Data Streams. In: Proc. of the 31st International Conference on Very Large Data Bases, pp. 1302–1305 (2005)
9. Ostfeld, A., Uber, J.G., Salomons, E.: Battle of water sensor networks: A design challenge for engineers and algorithms. In: WDSA (2006)
10. Hart, D.B., Klise, K.A., McKenna, S.A., Wilson, M.P.: CANARY User's Manual Version 4.1. Sandia National Laboratories. U.S. Environmental Protection Agency (2009)

Simdedup: A New Deduplication Scheme Based on Simhash

Wenbin Yao[1,2,3] and Pengdi Ye[1,2,3]

[1] Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
[2] Key Laboratory of Trustworthy Distributed Computing and Service(BUPT), Ministry of Education
[3] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China
yaowenbin@bupt.edu.cn, yepengdi@gmail.com

Abstract. Maintaining higher deduplication throughput with lower system overheads is a challenge for deduplication system in massive data storage environment. In this paper, a near-exact deduplication scheme named Simdedup is presented, which exploits file similarity and chunk locality to achieve the goal. Simdedup partitions a file object into several segments, and leverages similarity to find the most similar segments based on simhash algorithm. It exploits a deduplication cache in memory to store the chunk fingerprints of the most similar segment, which can raise the speed of the detection of redundant data. Simdedup needs few disk accesses for chunk lookup per file, which leads to a reasonable throughput. Experimental results show that Simdedup can perform better on system overheads and deduplication ratio than deduplication schemes that employ traits detection.

Keywords: Massive data storage, deduplication, similarity detection.

1 Introduction

With the rapid increase of massive data, deduplication technology has been introduced as an essential and critical component for massive storage systems. It not only can reduce storage space requirement, but also can improve the network throughput by eliminating the network transmission of redundant data.

Chunk-based deduplication is a commonly used deduplication technology, which divides data objects into fixed or variable length chunks. Typical chunk sizes are 4 to 8 KB. A hash fingerprint of each chunk is used to determine whether that chunk has been stored before. Chunks with the same fingerprint are assumed identical. New chunks are stored and references are updated for duplicate chunks. Chunk-based deduplication is very effective for backup workloads, which tend to be files that evolve slowly, mainly through small changes, additions, and deletions. However, unless some form of locality or similarity is exploited, chunk-based deduplication has to face the disk bottleneck problem. The total size of fingerprints required to detect duplicate chunks increases with the size of dataset, which may quickly overflows the RAM

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 79–88, 2013.

capacity and must be paged to disk. Moreover, hash fingerprint values are natively random, so the index of fingerprint cannot be cached effectively without locality, and it is not uncommon for nearly every index access to require a random disk access. This disk bottleneck severely limits deduplication throughput and increases the system overheads.

To provide high deduplication throughput and low system overheads, a new deduplication scheme named Simdedup is proposed for massive data storage environment in this paper, which leverages simhash [10] algorithm to achieve the goal. In Simdedup, only several disk accesses are needed for deduplicating a data object, so it can provide high deduplication throughput. Besides, the system overheads of Simdedup are relatively low due to the excellent features of simhash. As the length of simhash fingerprint is fixed and relatively small, it will not cost too much storage space. Meanwhile, comparing to the file chunking process, the time cost for calculating and comparing simhash fingerprint can nearly be ignored.

The rest of this paper is organized as follow. Section 2 introduces the background and related work. Section 3 presents the design and implementation of Simdedup. Section 4 conducts a series of experiments to compare the efficiency of Simdedup, deduplication scheme employs traits detection and the deduplication scheme in LBFS. The last section concludes our work.

2 Background and Related Work

LBFS [1] first proposes the content defined chunking algorithm to reduce transmission of redundant data. Venti [2] employs deduplication in an archival storage system and significantly reduces the storage space requirement. However, since no locality is exploited in the chunk fingerprint index accesses, these methods can result in frequent disk accesses for fingerprint lookups when the size of dataset increases, and thus limit deduplication throughput.

To deal with disk fingerprint lookup bottleneck problem, DDFS [3] proposes the idea of exploiting the backup stream locality to reduce accesses to on-disk index. Sparse Indexing [4] exploits the inherent backup stream locality by samples index for fingerprint lookup. Nevertheless, these locality-based approaches would produce unacceptably poor performance of deduplication in the case of the data streams with little or no locality [5].

As additional choices for deduplication, Aronovich [6] et al exploit the similarity of backup streams in mass deduplication systems, while Extreme Binning [5] exploits the file similarity for deduplication to apply to non-traditional backup workloads with low-locality. Similarity detection is a significant component for similarity-based deduplication system. There are several methods, such as shingle detection [7], bloom filter [8] and traits detection [9], which all use much smaller data fragments to represent the original data objects for similarity detection. However, the length of file features in shingle or bloom filter is related to the size of file, which means the bigger a file is, the longer fragment is needed to identify the file. The length of file features in traits is fixed and short, but its effectiveness lacks theoretical proof. Maintain fast

similarity detecting with low system overheads in large dataset is critical to deduplication schemes like these.

Simhash is a dimensionality reduction technique, which is proposed and proved by Charikar. It maps high-dimensional vectors to small-sized fingerprints. The input of simhash algorithm is an *n*-dimension vector, while the output is an *m*-bit fingerprint. Simhash has two main features as followed:

- The fingerprint of a data object is the simhash value of the corresponding vector of this data object.
- Other than information digest fingerprint, the simhash fingerprint guarantees that similar data objects have similar fingerprints.

These two features make simhash algorithm a suitable choice for detecting file similarity. Since the chunk fingerprints of files can be treated as vectors, it is easy to figure out simhash fingerprint of files by combining simhash algorithm and content based chunking algorithm. In additional, as the goal of this paper is to provide effective and fast similarity detection with low system overheads for deduplication system, simhash's low space and time complexity [11], as well as high precision [12], are quite appealing.

3 Design and Implementation of Simdedup

3.1 System Architecture Overview

As depicted in Fig.1, the Simdedup architecture is comprised of deduplication client and server. Major deduplication operations are completed on the client side. Deduplication server is the repository for backup data, which can be public cloud storage providers or private storage servers. In this way, users can switch storage service providers easily.

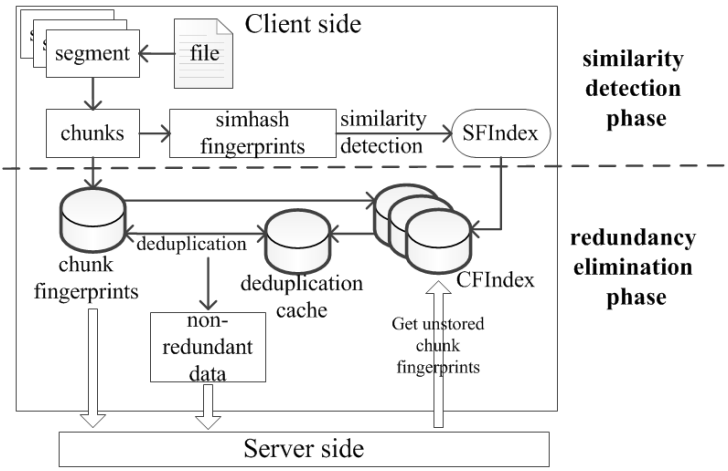


Fig. 1. The Simdedup system architecture

The original file is partitioned into several segments in Simdedup, so that a large file like archive file can be deduplicated as much smaller files to maintain low system overhead. Simdedup maintains two indexes of fingerprints in RAM and disk respectively to achieve excellent RAM economy and accelerate fingerprint lookups. The two indexes are simhash fingerprint index (SFIndex) and chunk fingerprint index (CFIndex).

SFIndex is responsible for maintaining similarity information of files. It stores all simhash fingerprints of segments. It is also responsible for finding the most similar segment to a new one. As the length of simhash fingerprint is fixed and relatively small, it will not cost too much storage space and can easily maintain in RAM. SFIndex is sent to the server side in cycles.

CFIndex takes charge of maintaining chunk locality for eliminating redundancy among files. It mainly offers three functions below:

- Store chunk fingerprints of each segment as a new file on disk respectively, so that only one disk access is needed for chunk lookup per segment.
- To keep the capacity of CFIndex away from growing with the dataset, LRU algorithm is employed to delete some rarely used chunk fingerprints.
- When the chunk fingerprints of the most similar segments are not stored in client side, obtain the corresponding data from the server side.

Based on the two indexes, Simdedup can exploit similarity and locality to achieve high duplicate elimination throughput at low system overheads. The whole deduplication scheme has two phases to exploits similarity and locality respectively, which are similarity detection phase and redundancy elimination phase.

Similarity detection phase mainly exploits file similarity. Each segment of a file is chunked by content defined chunking algorithm. The simhash fingerprints of segments can be figured out based on these chunks, which will be compared with the simhash fingerprints in SFIndex to detect similarity. In this phase, the most similar segments are found by similarity detection for exploiting chunk locality in the next phase.

Redundancy elimination phase exploits chunk locality based on the most similar segments to eliminate the redundant data. In order to leverage chunk locality, the chunk fingerprints of the most similar segments are read into memory from CFIndex to generate a cache for deduplication, while the chunk fingerprints of the new segment are calculated by SHA-1 hash algorithm. By checking these fingerprints, the duplicate chunks can be eliminated.

As the entire fingerprint lookups are almost completed in RAM, Simdedup can significantly reduce the disk accesses, and improve the deduplication throughput. Meanwhile, with the excellent features of simhash algorithm, Simdedup consumes very low system overheads.

3.2 Similarity Detection Phase

In this phase, the whole file object is partitioned into several segments. Every segment is chunked by content defined chunking algorithm. Then a group of chunks can be got, which will become the input elements of simhash algorithm. The choices of segment

size and average chunk size can be adjusted by the user's requirements like the backup throughput, duplicate elimination or the RAM usage. Fig.2 shows the process of figuring out *m*-bit simhash fingerprint, which can be described as followed:

1. Initialize an *m*-dimension vector *V*, every dimension of which is 0;
2. Every chunk of a segment is mapped to an *m*-bit signature by traditional hash algorithm. If the *i*-th bit of this signature is 1, then the *i*-th dimension of *V* will plus 1. Otherwise, it will minus 1;
3. Generate an *m*-bit simhash fingerprint *f* according to every dimension of vector *V*. If the *i*-th dimension of *V* is positive number, and then the *i*-th bit of *f* is 1. Otherwise, it will be 0.

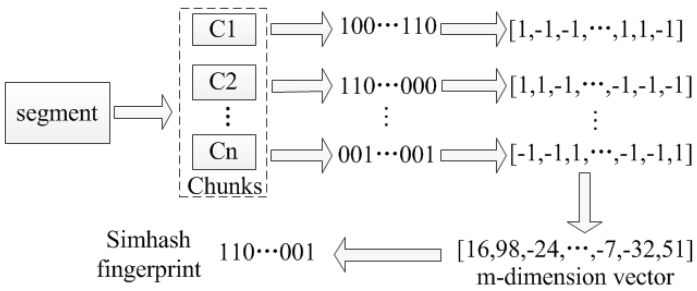


Fig. 2. Process of calculating simhash fingerprint

When simhash fingerprint of a segment is figured out, it will be compared to the fingerprints in SFIndex. The similarity of two segments can be got by calculating the hamming distance of their simhash fingerprints, which can be described by:

$$Sim(seg_1, seg_2)=H(f_1, f_2) \tag{1}$$

The smaller of hamming distance means the more similar of two segments, so the most similar segment has the smallest hamming distance to a new segment. Therefore, the most similar segment can be figure out by:

$$MostSimSeg(seg_{new}, Segs)=Min(Sim(seg_{new}, seg_i)),\ seg_i \in Segs \tag{2}$$

3.3 Redundancy Elimination Phase

After finding out the most similar segment, the redundancy elimination phase is started to eliminate the redundant chunk within files. As Fig.3 shows, this phase can be described by five major steps:

1. Calculate fingerprint for every chunk of the segment by SHA-1 hash algorithm.
2. When the most similar segment is got, the file that stores the corresponding chunk fingerprints is read into memory to generate a deduplication cache for redundancy elimination. If the file is not stored in CFIndex, fetch it from the server side.

3. Compare every chunk fingerprint with deduplication cache to eliminate deduplicated chunks. If two chunks have the same fingerprint, they are thought to be identical, and the one in the new segment will be discarded.
4. Organize entire chunk fingerprints of the new segment as a file on disk, which will be recorded in CFIndex to prepare for future segments.
5. Transfer the non-duplicate chunks and the file with chunk fingerprints of new segment to the server side.

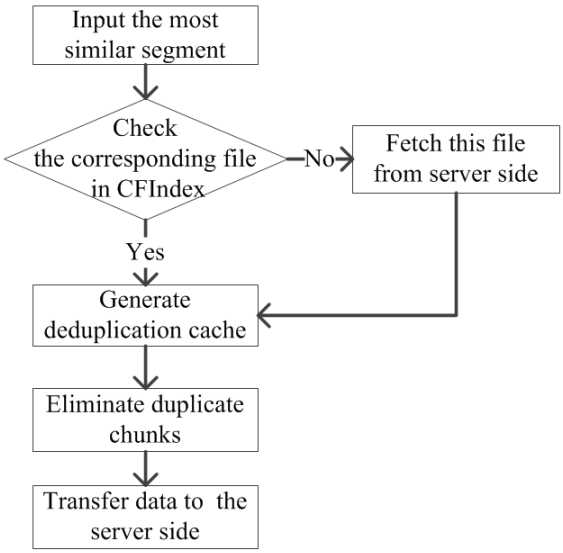


Fig. 3. Process of redundancy elimination phase

4 Experiment Results

A series of experiments are designed to analysis the efficiency of Simdedup on a computer with Intel core 2 Duo 2.93GHz processer, 2GB RAM and 320GB 7200RPM disk.

Traits detection based deduplication scheme with the same system architecture, which is named Traitsdedup, is implemented to compare deduplication ratio and system overheads with Simdedup. Database based deduplication scheme, which is proposed in LBFS, is implemented to compare deduplication throughput with Simdedup.

All these experiments are implemented in java and executed on a dataset of linux kernel source code packages with different versions. According to research [11], 64 bits simhash fingerprint is chosen here, while the size of traits is 96 bits based on research [9].

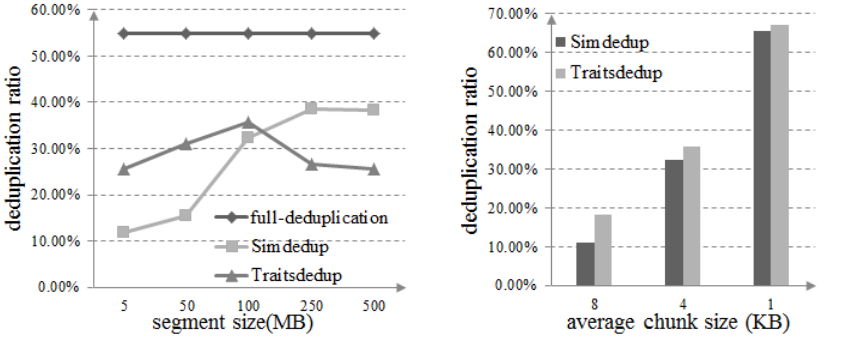
4.1 Deduplication Ratio

Two groups of deduplication ratio experiments are carried out in this section. A version of the linux package is chosen randomly as a new file that needs to be deduplicated. Fig.4(a) shows the results of experiments that are executed in different segment size and 4KB average chunk size, while Fig.4(b) shows the results of experiments that are executed in 100MB segment size and different average chunk size.

The deduplication ratio of Simdedup is less than Traitsdedup when the segment size is small in Fig.4(a). As the segment size grows, the deduplication ratio of Simdedup rises and exceeds even the best result of Traitsdedup, while the deduplication ratio of Traitsdedup drops after 100MB. These results indicate that when the segment size grows, simhash algorithm has better accuracy on similarity detection than traits detection algorithm.

Fig.4(b) indicates that the gap of deduplication ratio between Simdedup and Traitsdedup shrinks in 100MB segment size situation, when the average chunk size declines.

These results indicate the accuracy of similarity detection by simhash algorithm increases with chunk amount in one segment. Simhash algorithm has better accuracy to detect similarity than traits detection. Especially in massive data environment, bigger segment size means less segment amount and less simhash fingerprints, which makes SFIndex easier to fit to the RAM. In additional, 500MB segment with 4KB average chunk size generates about 2.5MB fingerprints of chunk. The file contains these fingerprint can be read into RAM in one time easily. The RAM overhead will still keep on a low level in this situation.



(a) compare by segment size (b) compare by average chunk size

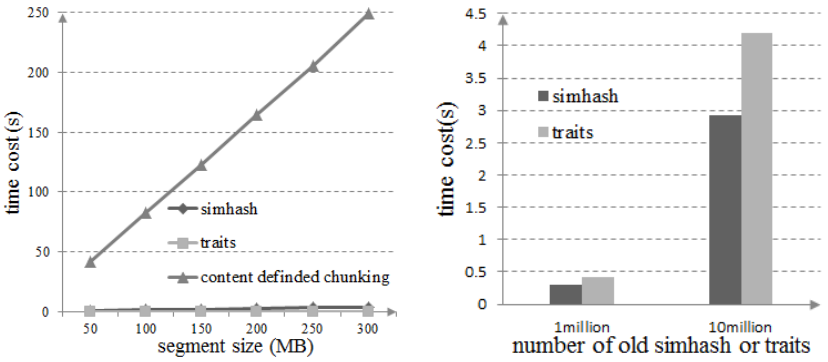
Fig. 4. Deduplication ratio compare

4.2 Extra Computation Overhead

Fig.5(a) shows the compare results of computation time cost in different size of segment among simhash, traits and content defined chunking algorithm. Statistics indicate that it takes liner time to complete the computation of these three algorithms. When the size of data grows, the time cost of these algorithms increases. Simhash need a little bit

more computation than traits, but comparing to content defined chunking algorithm, the time cost of calculating simhash can nearly be ignored.

Fig.5(b) reveals that simhash algorithm performs better than traits detection on compare time cost. When the size of fingerprints grows, the gap between the two algorithms become more obvious. Using shorter traits may eliminate this result, but the deduplication ratio will be affected because the accuracy of similarity detection declines when length of traits becomes shorter. Since calculating time cost is almost fixed when the segment size is decided and compare time cost grows with size of dataset, simhash algorithm may perform better on computation overhead in massive data storage environment.



(a) calculating time cost (b) compare time cost

Fig. 5. Computation overhead compare

4.3 Extra Storage Overhead

As 64 bits simhash fingerprint is used here, it means that only about 32MB storage space is needed to record four million simhash fingerprints for 1PB dataset if 250MB segment size is chosen. This size of fingerprints can be maintained in RAM easily, so that Simdedup can provide effective and fast similarity detection while nearly does not increase storage overheads. Since the typical size of traits detection algorithm is 96 bits, it may cost more storage space than Simdedup. Reduce the length of traits may cost the similar storage space with simhash, but the deduplication ratio will be deeply affected, so that simhash algorithm performs better on storage overhead than traits detection.

4.4 Deduplication Throughput

Since Simdedup and Traitsdedup both leverage similarity detection to eliminate duplicate chunks and share the same system architecture, their deduplication throughput are similar, so only Simdedup and the deduplication scheme in LBFS are compared in this section. As Fig.6 shows, the deduplication throughputs of Simdedup

with different segment size and same average chunk size have almost the same performance, but it increases when the average chunk size grows and the segment size keeps the same.

This may be because the number of chunks in one segment declines when the average chunk size grows, which decreases the size of files that stored the chunk fingerprints of segments. As a result, the time cost of reading these files drops, and then affects the deduplication throughput. However, all of the deduplication throughputs of Simdedup in different situations perform much better than the deduplication scheme in LBFS. When the size of dataset grows, the deduplication scheme in LBFS may perform even worse.

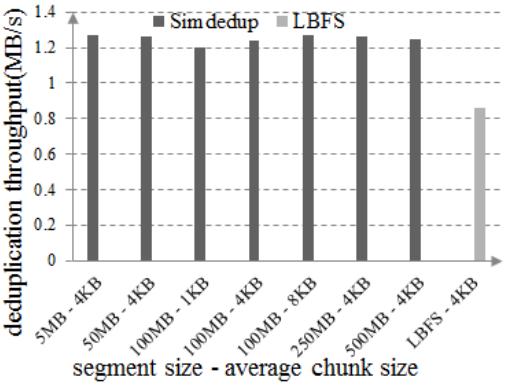


Fig. 6. Deduplication throughput

5 Conclusion

Simdedup is presented to maintain high deduplication throughput with low system overheads. Two phases deduplication scheme is used to reduce the disk access frequency and improve the deduplication throughput. It based on simhash algorithm to detect similarity at the first phase, which calculates simhash fingerprints for new file's segments and compare them with the old segments. At the second phase, the chunk fingerprints of the most similar segments will be read into memory as a cache to create chunk locality. Then compare chunk fingerprints with this cache, and eliminate duplicate chunks. Because the entire fingerprint lookups are almost completed in RAM, Simdedup can significantly reduce the disk access and provide excellent deduplication throughput. Experiments show that Simdedup performs much better on storage overheads and deduplication ratio than deduplication systems that exploit traits detection.

Acknowledgements. This work was supported by the National High Technology Research and Development Program ("863"Program) of China (2012AA012600) and the Fundamental Research Funds for the Central Universities (BUPT2011RCZJ16) and China Information Security Special Fund (NDRC).

References

1. Muthitacharoen, A., Chen, B., Mazieres, D.: A low-bandwidth network file system. In: Proceedings of the 18th ACM Symposium on Operating Systems Principles, pp. 174–187. ACM, New York (2001)
2. Quinlan, S., Dorward, S.: Venti: a new approach to archival storage. In: Proceedings of the FAST 2002 Conference on File and Storage Technologies, vol. 4 (2002)
3. Zhu, B., Li, K., Patterson, H.: Avoiding the disk bottleneck in the data domain deduplication file system. In: Proceedings of the 6th USENIX Conference on File and Storage Technologies, pp. 1–14. USENIX Association (2008)
4. Lillibridge, M., Eshghi, K., Bhagwat, D., Deolalikar, V., Trezise, G., Camble, P.: Sparse indexing: large scale, inline deduplication using sampling and locality. In: Proccedings of the 7th Conference on File and Storage Technologies, pp. 111–123. USENIX Association (2009)
5. Bhagwat, D., Eshghi, K., Long, D., Lillibridge, M.: Extreme binning: Scalable, parallel deduplication for chunk-based file backup. In: IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems, pp. 1–9. IEEE (2009)
6. Aronovich, L., Asher, R., Bachmat, E., Bitner, H., Hirsch, M., Klein, S.: The design of a similarity based deduplication system. In: Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference, pp. 1–14. ACM (2009)
7. Douglis, F., Iyengar, A.: Application-specific deltaencoding via resemblance detection. In: Proceedings of the 2003 USENIX Annual Technical Conference, San Antonio, Texas, pp. 113–126 (June 2003)
8. Broder, A.Z., Mitzenmacher, M.: Network applications of Bloom filters: A survey. Internet Mathematics 1(4), 485–509 (2003)
9. Teodosiu, D., Bjorner, N., Gurevich, Y., Manasse, M., Porkka, J.: Optimizing file replication over limited-bandwidth networks using remote differential compression, Technical Report MSR-TR-2006-157, Microsoft Research (November 2006)
10. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pp. 380–388 (2002)
11. Manku, G.S., Jain, A., Das Sarma, A.: Detecting near-duplicates for web crawling. In: Proceedings of the 16th International Conference on World Wide Web, pp. 141–150 (2007)
12. Henzinger, M.R.: Finding Near-Duplicate Web Pages: ALarge-Scale Evaluation of Algorithms. In: Proc. ACM SIGIR, pp. 284–291 (August 2006)

InfoMall: A Large-Scale Storage System for Web Archiving

Lian'en Huang, Jinping Li, and Xiaoming Li

Shenzhen Key Laboratory for Cloud Computing Application and Technology,
Peking University Shenzhen Graduate School, Shenzhen, Gongdong 518055, China
hle@net.pku.edu.cn, ljp0129@gmail.com, lxm@pku.edu.cn

Abstract. The World Wide Web is a fluid medium which means that Web pages or entire Web sites frequently change or disappear, often without leaving any trace. Considering the great value of the Web, it is quite necessary to archive the current Web for the future. In order to do this, a large-scale storage system is required. In this paper we propose such a system which is designed for storing the massive Web pages we have been collecting consistently since 2001. One significant feature of this collection of Web pages is that it is space-time dimensioned which means every Web page is attached with a URL and a time, while one URL is possible to contain lots of Web pages crawled at different times. Our system is designed that sorted Web pages are clustered and stored together by some degree of space-time granularity. As a result, users are able to retrieve effectively Web pages with URLs and times specified or batches of Web pages with URL ranges and time ranges specified.

Keywords: Web Archiving, Web Storage System, Web InfoMall.

1 Introduction

Web pages are unstable resources, which means sooner or later, they will disappear. Compared with that of traditional media like papers, the lifetime of Web pages is very short. Considering the vital role that the Web has been playing in modern society, it is our responsibility to preserve the current Web for future generations. Fortunately, the significance of preserving the Web has long been recognized since late 1990s, and various institutes have started to archive Web pages since then [1]. According to IIPC [2], more than 30 institutes have been involved in the effort of Web archiving.

Given the size of the World Wide Web, it is not hard to image what a huge collection of Web pages has been archived, and there are still much more Web pages waiting for archiving. And so here comes a big challenge of how to efficiently handle and process the huge number of Web pages, which unfortunately has seldom been discussed in the literature of Web archiving.

This paper presents InfoMall, a large-scale storage system designed for Web archiving. This is part of our effort to archive China World Wide Web, which is known as Web InfoMall [3]. We have been collecting Web pages since 2001 and as of now more than 7 billion pages have been collected and stored in our system.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 89–98, 2013.

2 Related Work

There are many applications who need to store lots of Web pages, for example search engines. The early implementation of Google described in [4] used ISAM-indexed files to store Web pages. Each page was compressed and stored one after another in a file, and a fixed width ISAM index was built which contains pointers to the Web pages. WebFountain [5] was another kind of application that requires management of large-scale Web pages. It was designed by IBM for data mining on billions of Web pages, which is also something we are interested in. In WebFountain, Web pages, along with other kinds of data, were stored across hundreds of servers in *bundles* each of which contained a few hundreds of records. To meet the critical demand of Web-scale data mining, WebFountain used high-performance hardwares to ensure efficient reading and writing.

The Stanford WebBase [6,7] gave an intensive study of storing Web pages from the view of academic society. In WebBase, Web pages were distributed with some policy across multiple severs and were stored in *buckets* each of which contained a number of Web pages. A series of techniques were discussed to improve the performance for different scenarios. However, the WebBase lacks the ability of storing different versions of a Web page, as is required in the task of Web archiving. When a new Web page arrives, it will replace the old one in the repository.

In recent years perhaps the most famous of general massive storage systems, which means they are not designed only or mainly for Web pages, is BigTable [8]. It is developed by Google and is normally based on GFS [9] to provide high reliability. Compared with traditional databases, BigTable provides a much simpler interface of key-value store and a looser data consistency model for efficiency. Later, Google also develops Megastore [10] on top of Bigtable to provide strong consistency guarantee like traditional databases. There are also similar systems like Dynamo [11] developed by Amazon, Windows Azure [12] developed by Microsoft and Cassandra [13].

A previous version of InfoMall was presented in [14]. A main drawback of the old system is that Web pages were stored without been sorted (by URL), so it was unable to retrieve a data set like "all pages of sina.com.cn".

3 Storage Model

The InfoMall storage system is designed for handling the massive Web pages that have been consistently collected for a long time, and will be continually collected in the future, from the World Wide Web by an Web archiving system. It is conceivable that the volume of the collection of Web pages will keep growing and growing, so it is important to store the Web pages in an orderly and extendable way.

The storage model of InfoMall is illustrated in Fig. 1. The input of the system is the Web pages crawled from the Web, which actually have a continuous long time span, measured by crawl-time. Here we only show a fragment of the Web pages, namely the ones crawled from Jun. to Aug. in 2008. The figure shows how this part of Web pages is stored in our system.

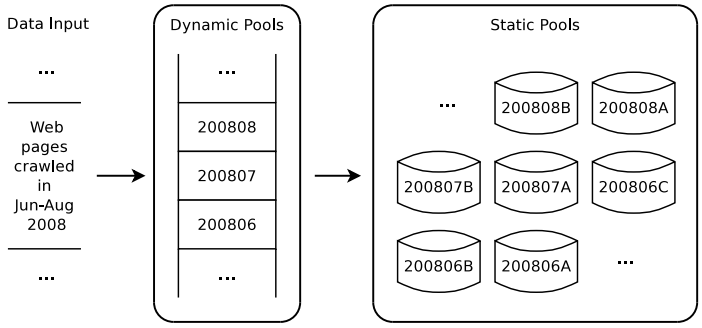


Fig. 1. The storage model of InfoMall

The basic storage unit we used is pool. A pool is essentially a database which mainly consists of a number of db-files each of which contains a large number of Web pages. Furthermore, a pool is constrained with a time range and, possibly, a URL range. That means a page can be put into the pool only if its crawl-time is within the time range and its URL is within the URL range. In our system we use month, a natural time unit, as the time constraint for a pool. As an example, the pools "200806", "200806A" shown in the figure, implied by their names, are constrained to the month June 2008, or they only contain Web pages crawled between 1st and 30th, June 2008.

As can be seen from the figure, there are two clusters of pools in our system: dynamic pools and static pools. One main difference between them is that in dynamics pools Web pages are sorted (by URL) only in each db-file while in static pools Web pages are sorted both within and between db-files. Or we can say dynamic pools are partially ordered while static pools are globally ordered. The other main difference lies in that dynamic pools are variable which means Web pages can be inserted into or removed from them, while static pools are invariable once they have been created.

To make clear how our system works, the following will give a detailed discuss on how pools operate and how Web pages can be accessed.

Pool Operations. Basically there are 6 operations designed for pools:

1. Inserting Web pages into a dynamic pool. Newly inserted Web pages will be collected into a newly created db-file until its size reaches some limit. Since the Web pages in a db-file should be sorted (by URL), the db-file should be in proper size so that it can be wholly loaded into the memory.
2. Turning the whole or a part of a dynamic pool into a new static pool. This is the process of making the Web pages from partially ordered to globally ordered. Actually it is done by reading the smallest one from the source pool.
3. Copying a pool into a new same-type pool. Note that this is not a process of simply copying files but a fully generation of a new pool, involving recreating all necessary files like index. Actually we require not simply copying files at

moving pools, and one reason for this is that we wish the system handle all the operations and leave logs there.
4. Inserting a static pool into a dynamic pool. This and the following 2 operations are normally used to reorganize static pools.
5. Merging several static pools into a new static pool.
6. Splitting a static pool to several new static pools.

Note that there is no deletion operation listed above. That is because for Web archiving there is no need to delete Web pages. However we do provide a mechanism for deleting Web pages: we create a filter which contains information about which pages should be deleted. A filter can be applied to the above pool operations so that when Web pages are moved from one pool to another, the unwanted ones will be filtered out.

Data Access Methods. We provide 2 data access methods with our system:

1. *unit access*: To retrieve a single (unit of) Web page from the system. It should be noted that a distinct URL is possible to contain a lot of Web pages of different versions, which means they were crawled at different times with the same URL. So besides a URL, a time must be specified, and the system will search a time-span extended by the requested time. If multiple versions of Web pages have been found, the one whose time is nearest to the requested time will be returned.
2. *range access*: To retrieve a batch of Web pages specified by a URL range and a time range. In our system we store URLs in a slightly modified form. For example, the URL "http://www.foo.com/index.html" is modified as "http://com.foo.www/index.html". So it is possible to retrieve a whole website like "foo.com" which might contain hosts "a.foo.com", "b.foo.com", and so on. Expressions like "site(foo.com)", "host(www.foo.com)", "[min-URL, max-URL]" can be used to specify a URL range.

Theoretically, the process of retrieving Web pages can be explained as this: The system first calculates which pools are involved and sends queries to them. When it receives results from the pools, it then merges the results before sending them to the users. The actual procedure depends on how it is implemented.

4 System Implementation

4.1 System Architecture

Our system is designed to run on a cluster of ordinary linux servers, each of which typically has a relatively big capacity of storage, compared with normally-configured servers. The hardware machines are supposed to be much more stable than personal computers, and so a software system like GFS [9] which provides high reliability is not critically necessary. Maintainability and performance are the two key concerns in designing our system, while reliability is also largely considered.

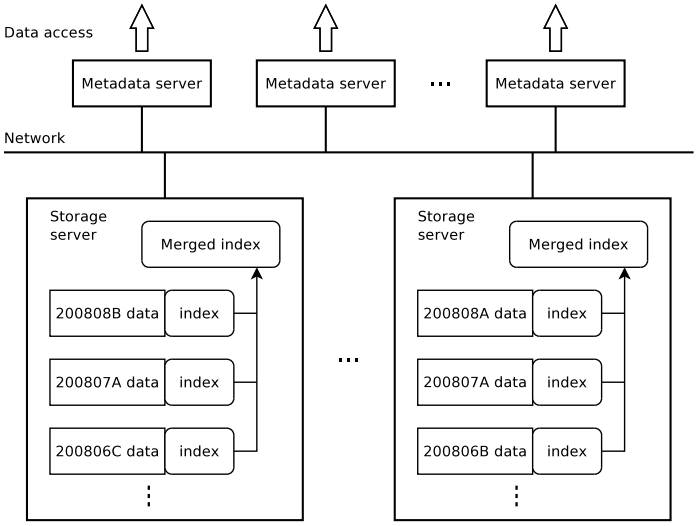


Fig. 2. The architecture of InfoMall

The architecture of InfoMall is shown in Fig. 2. As can be seen from the figure, there are two kinds of servers in our system: metadata servers and storage servers. Metadata servers manage metadata information which is periodically collected from storage servers and provide data access services to users, while storage servers contain lots of pools and create indexes to improve data access performance. There are also two types of storage servers in accordance to the two types of pools: dynamic ones and static ones. It is self-explained that a dynamic storage server contains a number of dynamic pools while a static storage server contains a number of static pools. Except for that difference, the two types of storage servers work very similarly, and so in the figure we only show static storage servers for simplicity. Actually a storage server is configured to be a dynamic one or a static one.

Pools in the storage servers are generated by the pool operations described in the previous section. It should be mentioned that pools can be duplicated. For example, the figure shows two identical pools named "200807A". The duplicates are located in different servers so that if one server is damaged the data still survives in the other server. This mechanism is greatly useful to keep the valuable Web pages we have collected in more than 10 years in safety.

Unlike BigTable [8], which use a master node to store the root tablet information, there is no master node or control center in our system. Each server is identified by an IP address (combined with a port), and each server keeps a list of other servers which can be customized. There is possibility that some storage servers are dead, but the other servers still alive will continue to provide service. A metadata server will periodically detect whether a storage server is dead, and collect metadata information from it if it is not dead. When a query arrives, the

metadata server will parse the query, send its queries to storage servers involved, and merge the results before sending them to the user.

To improve the performance of searching for Web pages by URLs, we build a global index for each storage server. Note that each pool has its own index, so what we need is to merge the indexes of all pools. When a new pool is added to the server, or an old pool is removed from the server, the global index then shall be rebuilt. A background process is dedicated to this purpose, however it actually rebuilds the index by creating a new one and removing the old one. With the global index built, a pool's index is accessed only if it has not been included in the global index or it has changed (if the pool is a dynamic pool). And as a result, the performance of searching is significantly improved since there are normally quite a number of pools in a storage server.

4.2 Metadata Management

Metadata is a kind of Information that helps us quickly locate the Web pages we need. As is mentioned above, pool is the basic storage unit of our system, and so pool is also the basic unit of metadata. For each pool, when requested by a metadata server, a little information is sent to the metadata server, which includes:

1. The URL range of the pool which is normally composed by the minimal URL and the maximal URL.
2. The time range of the pool which is normally the time-span of a natural month.
3. The number or sequence range of its Web pages.

The metadata server then will maintain a list of such information and its index. With this information, the metadata server is able to know where to get what it need for answering users' requests and make an optimized processing. For example, if a user is requesting a Web page crawled at June 2008, the request will be relayed to the minimally-needed storage servers that contain pools "200806", "200806A", "200806B" and "200806C".

However, as a drawback of not maintaining the metadata consistently, there is a possible problem of data inconsistency: when a user is using the *range access* method to retrieve a batch of Web pages, it is possible that he will encounter the problem of data inconsistency, which means, he might miss some Web pages that he is supposed to retrieve. There are two cases that could cause the problem. The first case is that during the time of retrieval the whole or a part of a dynamic pool has been turned into a new static pool and removed from the dynamic pool, so he will miss some Web pages if he is going to visit the dynamic pool later. The second case is that some storage servers are dead, and so the metadata server will miss their data during their death. As a possible solution to this problem, the metadata server can keep a snapshot of metadata at the beginning of retrievals, and then inform users if it has detected the possibilities of data inconsistency so that users can further retrieve the missed Web pages or just choose to re-retrieve. Moreover, it is normally tolerable for most Web archiving related applications to miss some Web pages.

4.3 Data Redundancy

The safety of data is a key concern in designing our system, simply because the purpose of Web archiving is to preserve the Web pages for the future. If the Web pages are lost, then what we have done turns into nothing. Of course we keep off-line backups outside the system, but we still always want to keep the system safe, and data redundancy is the answer to this concern.

In the current system we keep at least 2 duplicates for each pool, each of which is located in different storage servers. However, we apply different polices to the two types of pools respectively. For static pools, we define that if 2 static pools share the same name, then they shall be duplicates or they shall be identical, and we can put the 2 static pools to any 2 servers we choose. For dynamic pools, since we can not assure 2 dynamic pools share the same name to be identical all the time, we duplicate them in the server level. That is, we maintain 2 dynamic storage servers in the same way to make them duplicate servers. If we insert some Web pages into one server, then we shall also insert the same Web pages into the other. If the Web pages are inserted in the same order, then the pools in the 2 dynamic storage servers are supposed to be created identically. Also it is possible to synchronize the 2 dynamic storage servers in case of errors. Although the 2 dynamic storage servers are not strictly consistent to each other, we believe this is tolerable for most applications.

There are also other kinds of data redundancy. For example, RAID (Redundant Arrays of Inexpensive Disks) provides data redundancy on the hardware level, which is widely used in our servers. GFS [9] provides data redundancy on the level of file system. While our system provide data redundancy on the level of database. Since there are no conflicts among them, it is possible to combine the three techniques to achieve rather high reliability.

5 Performance Evaluation

Our system is currently running online with a website [3] on the CERNET[1] and the Web pages stored can be visited and retrieved from the website[2]. The current system is running on a cluster of 19 ordinary servers which were bought in different years: 3 bought in 2006, 8 bought in 2010 and 8 bought in 2012. The 3 old servers are dedicated as metadata servers, while others are used as storage servers. As of now, the system has stored a collection of more than 7 billion Web pages.

5.1 Performance of Data Input

The aim of this evaluation is to test how fast our system can import Web pages from outside. The Web pages were crawled by our crawler from the China Web

[1] China Education and Research Network.

[2] Due to the high fee for abroad visit on CERNET, the website is currently limited to domestic visit.

Table 1. Time used for importing 50 million Web pages

system used	the current system	a new empty system
time consumed	657 minutes	613 minutes

since 2001, and they were compressed and saved by the crawler as files using a standard format we have defined for our system.

First, there is a result comes from the first 3 billion Web pages imported into the system. The 3 billion Web pages were originally stored in a previous version of InfoMall, then when the current system was built, they were reimported to the current system. According to the logs, it takes about 27 days to import the 3 billion pages, namely more than 100 million per day.

Second, we conduct an experiment with a new set of 50 million Web pages. We use the current system (more than 7 billion pages have been stored) and a new empty system to import the data respectively, and the time consumed is shown in Table 1.

So both results tell that our system is capable of importing more than 100 million Web pages per day. We believe it is quite enough for a Web archiving project.

5.2 Performance of Data Access

There are two types of data access for evaluation: *unit access* and *range access*. To evaluate the performance of *unit access*, we first selected 10000 most frequently visited URLs from the logs of the TianWang search engine [15] which are also maintained by us. Then the URLs were combined with randomly generated times between Jan. 2002 and Dec. 2008 to retrieve Web pages from our system. We varied the number of clients from 20 to 100 to test how immediately our system responds under different workloads.

The result is shown in Fig. 3. The throughput of TotalFetch is roughly 220 times per second while the throughput of GoodFetch, which means the URL was found and a Web page was returned to the client for a given fetch, is about 45 times per second. Also the mean-response-time is about 400 milliseconds when

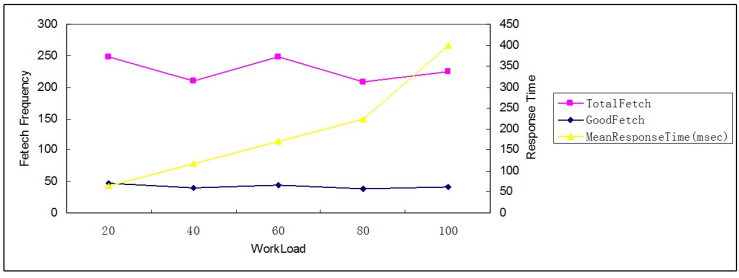


Fig. 3. Performance of unit access

Table 2. Performance of range access

number of clients	1	2	3	4	5
download speed (MB/s)	6.8	9.2	10.8	10.7	11.0

the number of clients reached 100. Moreover, it should be noted that our system was still providing Web access service on the Internet during the experiment.

To evaluate the performance of *range access*, we first selected 30 popular websites, for example sina.com.cn, sohu.com. Then the websites were combined with randomly selected time-ranges, for example Jan.-Dec. 2002, to retrieve batches of Web pages from our system. We varied the number of clients from 1 to 5, which is not a big value since we do not mean to provide this kind of access method publicly and widely.

The result is shown in Table 2. Note that the Web pages were compressed with gzip [16] before returned to clients, and that the network devices we used are all 100-Mbps ethernets, which have a speed limit of about 12MB/s. If equipped with better devices, we believe that our system is capable of providing higher speed of data access.

6 Summary

This paper describes InfoMall, a large-scale storage system designed for Web archiving. In this system, sorted or partially sorted Web pages are clustered and stored together into pools by some degree of space-time granularity, namely each cluster or pool is constrained by a URL range and a time range. As a result it is able to effectively retrieve Web pages with URLs and times specified or batches of Web pages with URL ranges and time ranges specified. With techniques like maintaining server-level indexes and applying data redundancy, the system also achieves good performance and high reliability. A running system has been set up and experiments have been conducted which demonstrate the effectiveness of our system.

Acknowledgments. We thank the anonymous reviewers for their valuable and constructive comments. This work is financially supported by NSFC under grant No. 61272340.

References

1. Toyoda, M., Kitsuregawa, M.: The History of Web Archiving. Proceedings of the IEEE 100, 1141–1143 (2012)
2. The International Internet Preservation Consortium, `http://www.netpreserve.org`
3. The Web InfoMall, `http://www.infomall.cn`

4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: 7th World Wde Web Conference, Brisbane, Australia, pp. 107–117 (1998)
5. Gruhl, D., Chavet, L., Gibson, D., et al.: How to build a WebFountain: An architecture for very large-scale text analytics. IBM Systems Journal 43(1), 64–77 (2004)
6. Hirai, J., Raghavan, S., Garcia-Molina, H., et al.: WebBase: A repository of Web pages. In: 9th World Wde Web Conference, Amsterdam, The Netherlands, pp. 277–293 (2000)
7. Cho, J., Garcia-Molina, H., Haveliwala, T., et al.: Stanford WebBase Components and Applications. ACM Transactions on Internet Technology (TOIT) 6(2), 153–186 (2006)
8. Chang, F., Dean, J., Ghemawat, S., et al.: Bigtable: A distributed storage system for structured data. In: 7th USENIX Symposium on Operating Systems Design and Implementation, Seattle, USA, pp. 205–218 (2006)
9. Ghemawat, S., Gobioff, H., Leung, S.-T.: The google file system. In: 19th ACM Symposium on Operating Systems Principles, New York, USA, pp. 29–43 (2003)
10. Baker, J., Bond, C., Corbett, J.C., et al.: Megastore: Providing Scalable, Highly Available Storage for Interactive Services. In: 5th Biennial Conference on Innovative Data Systems Research (CIDR 2011), Asilomar, California, USA, pp. 223–234 (2011)
11. DeCandia, G., Hastorun, D., Jampani, M., et al.: Dynamo: Amazon's Highly Available Key-value Store. In: 21st ACM Symposium on Operating Systems Principles, Stevenson, Washington, USA, pp. 205–220 (2007)
12. Calder, B., Wang, J., Ogus, A., et al.: Windows Azure Storage: A Highly Available Cloud Storage Service with Strong Consistency. In: 23rd ACM Symposium on Operating Systems Principles, Cascais, Portugal, pp. 143–157 (2011)
13. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review 44(2), 35–40 (2010)
14. Huang, L., Yan, H., Li, X.: Engineering of Web InfoMall: The Chinese Web Archive. In: World Engineers Convention 2004, Shanghai, China, vol. A, pp. 217–222 (2004)
15. The TianWang Search Engine, `http://e.pku.edu.cn`
16. GNU zip, `http://www.gzip.org`

Sequential Record Based Compression for Massive Image Storage in Database*

Ziyun Ma[1], Xiaonian Wang[1], Ping Jiang[2], Jiajun Jin[1], and Silu Guo[1]

[1] Department of Information and Control, Tongji University, Shanghai 200092, China
[2] Department of Computer Science, University of Hull, HU6 7RX, UK

Abstract. With the development of automatic management for industrial manufactory, the applications involving computer-vision and pattern recognition are widely used. The advantages of these modern methods using database to store a large sequence of images for helping management have been widely recognized. However, the storage of massive images into a database may demand a large memory space and cause a slow access speed. To increase the utilization rate of storage space and improve the performance of the database, this paper proposes an image compression scheme which is learnt from video compression to remove temporal and spatial redundancy in the image sequence. The proposed scheme not only alleviates the above issues associated with image storage but also keeps the basic database operations valid to image access, such as, insert, delete and update. At the end of this paper, we use the result to solve the optimizing storage problem successfully for a packaging machine which is computer-vision based quality monitoring.

Keywords: Database, Image Storage, Video Compression, Temporal and Spatial Redundancy.

1 Introduction

As the development of techniques for image acquisition and processing, video and images have become commonplace in industries and our daily life, such as in manufacturing, security, and traffic surveillance. Saving video or a sequence of continuous images usually demands a huge amount of storage space whilst individual image frames need to synchronize with other information, such as time stamps and sensor events, for their applications. Generally, there are two modes to save images[1], saving in a file system or in a database. The file mode adopts FTP (File Transfer Protocol) to store digital images in a FTP server. Sequential images, organized in file folders, can be saved in order hierarchically. Then clients can access them by finding certain index rules in connection with the folder structure. On the other hand, the database mode stores images as BLOBs in a database[2]. The database mode is more commonly used for applications with a large amount of images. However, the current methods usually

* Supported by Major State Basic Research Development Program: 91120308.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 99–107, 2013.

store the whole images or their compressed counterparts into a database, which result in a large memory usage and a low access speed.

To solve the problems associated with the transmission and storage of video or sequential images, video/image compression techniques have been developed. Some of which compress a single still image and remove the spatial redundancy of an image on the promise of a given reconstruction quality, such as the JPEG image compression algorithm[3]. On the other hand, compressing temporal redundancy between neighboring frames is also developed for video or sequential images, which considers the correlation between the current frame and adjacent frames. Simultaneous compression in both spatial and temporal domains can achieve a higher compress ratio than in either domain alone, which has been adopted in MPEG-1[4], MPEG-2[5], MPEG-4[6], and H.264[7]. However, they take a video or a sequence of images as a single entity. The frame-wise metadata relations with other events, that often need to be maintained in many applications, are difficult to be managed by using the standard compression algorithms. Video or sequential image compression in database needs to be developed.

When saving a sequence of sampled images in a database, the increase of rows in a relational database indicates passed time. This paper introduces compression operations for the database storage to remove the redundancy of the sequential images in both spatial domain and temporal domain. The proposed method can be described as compressing the redundancy in spatial domain of a single frame first and then removing the redundancy between neighboring frames along the direction of rows in a database. Working in this way, we can effectively reduce the storage space of a sequence of images, increase the accessing speed and maintain metadata relations to the resolution of a single frame, for example to alarm an abnormal image associated with events from sensors.

This paper is organized as follows: Section 2 presented the methods of sequential image compression, including encoding and decoding methods. A detailed solution to sequential image storage in database is given in Section 3. Section 4 presents experimental results on a packaging machine for image based product quality control. We summarize and conclude this study in section 5.

2 Video Compression for Sequential Images in Database

To compress video images in spatial domain and temporal domain, the video frames are processed by intra-frame compression and inter-frame compression. The intra-frame compression is used to save reference frames, which are taken as references to code the subsequent images. On the other hand, the inter-frame compression encodes the subsequent images using motion estimation from a reference frame[8]. Motion estimation has been considered as an effective way to eliminate temporal redundancy in moving images[9]. In the encoding process of the inter-frame compression, motion vectors from a reference frame is predicted first and then the residuals between the current image blocks and the predicted ones are obtained. The residuals are then transformed, quantized after the encoding.

In video compression, block matching motion estimation (BMME) and compensation are essential in several video-coding standards, such as MPEG-1/2/4 and ITU-T

H.261/263/263+. The best match is used as the predictor of the block in the current frame, whereas the displacement between the two blocks defines a motion vector (MV). Every block in the current frame can be expressed as a motion vector and a residuals block which is defined as the difference between the actual block and the predicted block [10].

As the running time increases, the database could occupy a huge amount of memory space. This paper modifies the standard video compression scheme to be more suitable for storing a sequence of images in a database for real-time applications, which includes the selection rules of reference frames and coded frames and the generation of motion vectors[11-12]. The proposed scheme saves some images as I-frames (the reference frames) and saves the others as P-frames (the coded frames) with forward-prediction only. The motion vectors are defined as absolute motion vectors between coded frames and reference frames instead of the relative motion vectors between neighboring frames. Furthermore, the images of a reference frame are compressed by using JPEG standard and the changes of a coded frame are estimated to calculate the absolute motion vector between the coded frame and the reference frame and the residuals between the coded frame and the predicted frame. After applying Huffman encoding on the motion vectors and apply transformation and quantization on the residuals, we save them into a database. The flow diagram of the compression process for a reference frame is shown as Fig.1.

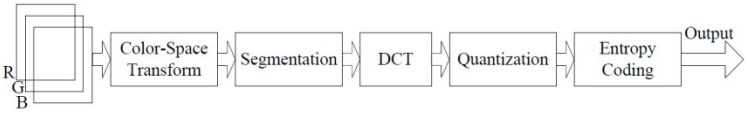


Fig. 1. I-frame compression Diagram

The flow diagram of the compression process for a coded frame is shown as Fig.2.

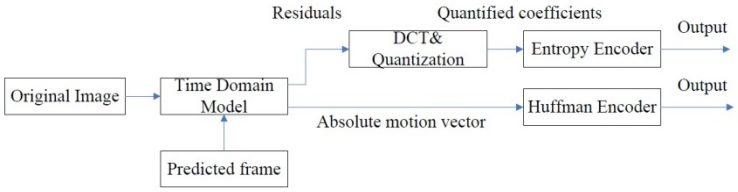


Fig. 2. P-frame compression Diagram

Furthermore, we define a decision function as the criteria to judge if a frame should be intra-frames encoded or be inter-frames encoded. We set the threshold of this function as N. If $\sum_i \sum_j D_{i,j}$is not less than N, the frame is considered as a reference frame and is encoded with intra-frames compression as in Fig.1. Otherwise, the frame is a coded frame using the inter-frame encoding as in Fig.2.

3 Database Storage and Operations

Section 2 draws the idea of video compression for saving sequential images in a database, where every image is saved in a row of the database. It removes the redundancy in spatial and temporal domains to effectively reduce the size of the database.

Table 1. Column Definition of *Tab.Images*

Column	Type	Attribute
[ImageID]	[bigint]	Primary Key, Not Null
[KeyBLOB]	[image]	Not Null
[Reference]	[bigint]	Not Null

We define the original database as Tab.User. In this paper, for the enhancement of database performances, we add a datasheet named Tab.Images which is imperceptible to users to save reference frames. Tab.Images stores the results of intra-compression. We design three columns in Tab.Images as shown in Table 1, ImageID, KeyBLOB and Reference, where ImageID is to store the index of records as a unique identification of each row, KeyBLOB is to store the reference frames' complete image information after intra-compression and Reference is to store the number of references by Tab.User.

We suppose that BLOB is the column which stores the image information in Tab.User. In order to keep an identical structure for Tab.User, we change the content of BLOB according to the type of a frame. BLOB keeps the first 64 bits for ImageID before the combined information, which links the two tables.

Basic database operations for the proposed scheme are developed as follows.

1. Insert

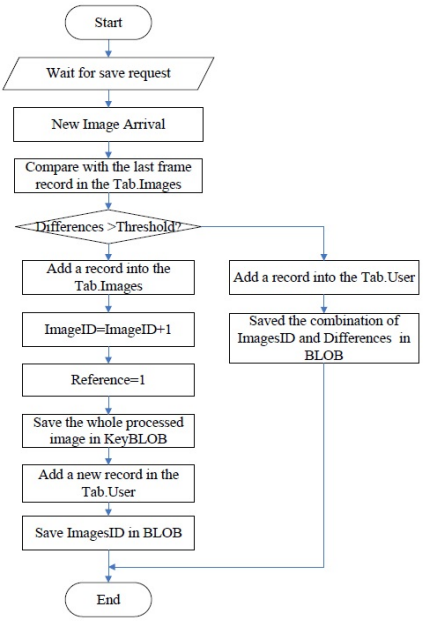


Fig. 3. Flow chart of saving new images

Instead of saving every image into the database, the first image is chosen as the reference frame, which is used to compare with the following images to get their difference (motion vector and residuals). To add a new record, the program will read the last record's image content in Tab.Images and make a comparison with the incoming image. The related reference frame's ImageID and the difference will be combined together and saved in the column BLOB in Tab.User if the new image is judged to be a coded frame of this reference frame. For data integrity, the last record's Reference attribute will be updated by increasing one in Tab.Images. Otherwise, the new image will be saved as a reference frame, i.e. a new record will be added into the Tab.Images, and where ImageID attribute will increase one to index a new reference image. The whole image information compressed in the JPEG format is saved in column KeyBLOB. And the column Reference is set to one. Meanwhile, a record is added into the Tab.User, which stores the reference frame's ImageID in column BLOB. The flow chart of saving a new image is shown as Fig.3.

2. Select

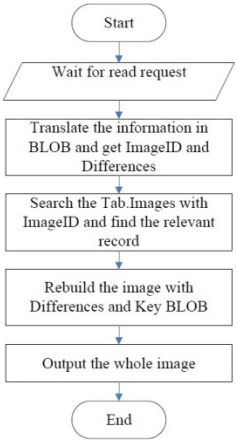


Fig. 4. Flow chart of reading records

When a user intends to read a record in Tab.User, the program will search Tab.Images based on the reference frame's ImageID saved in Tab.User and get the KeyBLOB in both tables. Then it will be decoded by combining the content of difference, which is stored in Tab.User's BLOB after the first 64 bits, with the reference frame's content in Tab.Images's KeyBLOB using the method described in section 2. After the decoding, the program will output the reconstructed image to the user. The flow chart of reading an image record is shown in the following Fig.4.

3. Delete

Because every reference frame is saved in Tab.Images, any record in Tab.User is allowed to be deleted. The value of column Reference in Tab.Images will be reduced after the deletion of a coded frame in Tab.User.

In order to keep the integrity of the whole sequence, we cannot delete the records in Tab.Images straightforwardly. We need to check whether the value in column Reference is one before delete the record. Only the independent reference frames can be directly deleted in the Tab.Images. The independent reference frames are those frames referenced by Tab.User only once. For those records had been referenced, we have to keep them in Tab.Images.

The process of deleting a record is shown as following Fig.5.

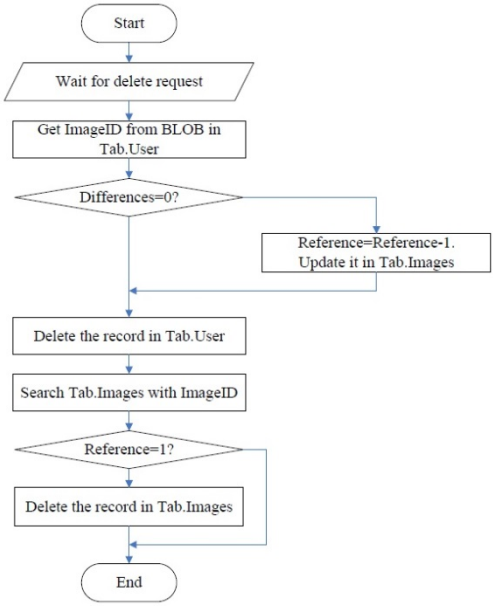


Fig. 5. Flow chart of delete records

4. Update

All the records of the image information in the Tab.User can be revised. But before updating a record in Tab.Images, the validity needs to be checked in order to avoid damaging other images in the database. If the record to be revised is a coded frame, the program will keep the relevant reference frame's ImageID, re-calculate the differences and update the column BLOB in Tab.Images. If it is an independent reference frame, the information saved in column KeyBLOB of the changing record can be altered directly. If it is a depended reference frame, the updated contents should be treated as insert a new reference frame. And the original reference frame should be kept in Tab.Images. Those coded frames related the reference frames have no need be re-calculated. The flow chart of updating is shown as Fig.6.

In summary, the proposed operations added on the top of database commands can ensure the integrity of sequential images and the completeness of database. There is no new column appears in Tab.User, so even if a user sets the other columns as the index and rearranges the Tab.User with it, the program can still find the correlated reference

frame by searching the ImageID saved in KeyBLOB for the record and reconstruct the image by the approach introduced above.

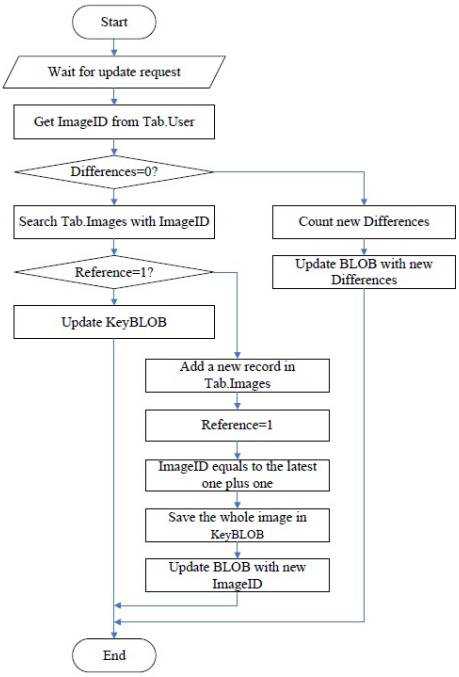


Fig. 6. Flow chart of updating record image

We take the cost of running time to reduce storage space. But all the operations are implemented as background processes in server without user's awareness.

4 Experimental Results

The proposed method was developed for an industrial project to control quality of an automatic packaging machine in a production line using a camera. In the process of packaging, due to static electricity and displacement of the stacker's push board, it is likely to cause missing or dislocation of cigarettes. With a computer-vision system to monitor the packaging process, the production quality can be promised. This system is mainly for tracking production quality by recording sequential images and the associated test results during packaging process into a database, which demands a large amount of memory space.

Fig.7 shows four boxes' images during the continuous packaging process. It can be seen from the figure that those images are similar with only slight translation and rotation. The proposed method can effectively reduce the memory space to save the day-to-day production images in a database.

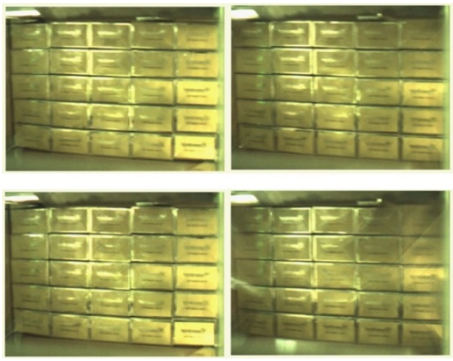


Fig. 7. Images of packing process

In the original system, only intra-frame compression is used, where individual images in JPEG format are inserted into the database. The size of each JPEG color picture with a resolution of 640*480 is about 100kBytes. The storage space needs about 720Mbytes every day. It challenges the storage space.In the final computer-vision based quality control system, the proposed method is used to improve the efficiency of the database. The storage space is reduced to about 240Mbytes when the PSNR is 35 and reduced to about 100Mbytes when PSNR is 25.

The experimental results show that the solution proposed in this paper can save the storage space for database applications involving sequential images, but it takes extra time to compress images.

5 Conclusion

In this paper, we present a new solution to save a great amount of images into a database based on video compression, in which the redundancy in time domain between consecutive images and the redundancy in space domain in a single frame are removed. On the permission of database's normal operations and its completeness, this solution can largely reduce the size of storage space. Especially, it is suitable for saving sequential images that have mild changes between neighboring images.

The proposed scheme can be implemented as a middleware between images based applications and a database. From a user's point of view, accessing the images in the database exhibits no difference from the normal database operations and all encoding and decoding processes are hidden from the user. This paper used a simple compression algorithm for efficient images storage in a database. More sophisticated algorithms, such as compression based on objects[13], compression based on segmentation[14], can be adopted for performance enhancement but without any influence on database operations.

References

1. Yang, N., Shen, Q., Xie, J.: Technology Research of Image Access in SQL Server Database. Journal of Nanjing Xiaozhuang University 5, 82–86 (2010)
2. Liu, X.: Study on the strategies and methods for the picture storage in the database. SCI/Tech Information Development&Economy 15, 206–207 (2005)
3. Wallace, G.K.: The JPEG Still Picture Compression Standard. Communications of the ACM - Special Issue on Digital Multimedia Systems 34, 30–44 (1991)
4. ISO/IEC/JTCI/SC29/WGll, ISO/IEC, MPEG-1 Committee Draft[S], CD11172: Information Technology (1991)
5. ISO/IEC/JTCI/SC29/WGll: ISO/IEC, MPEG-2 Committee Draft[S], CDl3818: Information Technology (1993)
6. ISO/IEC/JTC1/SC29/WG11: MPEG-4. Overview, Doc. N3156 (1999)
7. Wiegand, T., et al.: Overview of the H.264/AVC Video Coding Standard. IEEE Trans. Circuits Syst. Video Technol. 13, 560–576 (2003)
8. Ang, P.H., Ruetz, P.A., Auld, D.: Video Compression Makes Big Gains. IEEE Spectrum 28, 16–19 (1991)
9. Yao, W., Ostermann, J., Zhang, Y.: Video Processing and Communications, pp. 111–120. Pearson Education (2003)
10. Tourapis, A.M., Au, O.C., Liou, M.L.: Predictive Motion Vector Field Adaptive Search Technique(PMVFAST)—Enhancing Block Based Motion Estimation. In: Proc. Visual Communications and Image Processing 2001, VCIP 2001 (2001)
11. Rijkse, K.: H.263: Video Coding for Low-Bit-Rate Communication. IEEE Commun. Mag. 34, 42–45 (1996)
12. Yang, J., Chen, X.: Research of Image Compression Technology Based on MPEG-4. In: IEEE 3rd International Conference on Communication Software and Networks, ICCSN (2011)
13. Lu, B., Wang, S.: Fast Mode Decision Method Based on Mode Grouping For H.264/AVC. Computer Application and Software 25, 120–125 (2008)
14. Chen, T.-C., Chen, Y.-H., Tsai, S.-F.: Fast Algorithm and Architecture Design of Low Power Integer Motion Estimation for H.264/AVC. IEEE Trans. Circuits Syst. Video Technol. 17, 232–238 (2007)

Research of Network Coding Data Collection Mechanism Based on the Rough Routing in Wireless Multi-hop Network

Jian Wan[1], Ligang He[1], Wei Zhang[1,⋆], Jie Huang[1], Mingbiao Li[2], Jilin Zhang[1], and Nan Chen[3]

[1] School of Computer and Technology, Hangzhou Dianzi University, Hangzhou 310037, China
[2] Wenzhou University, Wenzhou 325035, China
[3] Information Centre, Zhejiang Tobacco Corporation
{wanjian,huangjie,jilin.zhang}@hdu.edu.cn, hlg.06@163.com, magherozhw@gmail.com, jsjlmb@wzu.edu.cn, cnasd715@zju.edu.cn

Abstract. In wireless sensor networks, network coding technology can enhance the data persistence. The classic wireless sensor network distributed storage algorithm LTCDS-1 may cause serious cliff effect in decoding process. To address this issue, this paper proposed a strategy BRRCD which is based on the rough routing for collecting data in packet-degree increments in wireless sensor network coding. A collector broadcasts a signal to form a layered network. Meanwhile, each node records the neighbors which are upper one layer of the network of itself, as the optional paths to reach the collector. The experiment results show that the BRRCD algorithm restrains the cliff effect in the extent.

Keywords: wireless sensor networks, network coding, rough routing.

1 Introduction

Wireless ad-hoc network is usually characterized with self-organizing, multi-hop, dynamic topology and energy-resources restriction. It can sense and obtain a large number of detailed and reliable data in coverage area of the network, and data is sent to the sink node with wireless multi-hop manner. There is a broad application prospects in the field of military defense, health care, environmental monitoring and so on. Sensor node has the characteristics of small size and limited energy, limited computing power, limited storage capacity, and limited communication capabilities. It is generally deployed in the disaster environment, such as: earthquake, flood, fire and so on. Under the influence of such harsh environment, the availability of sensor node can be greatly affected [1]. Therefore, under such environment, it is important to find a mechanism to improve the speed of distributing data and recovery efficiency of data decoding, while the

⋆ Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 108–117, 2013.

sensed data can be distributed in the damaged network with high robustness by means of network coding.

The data storage and collection strategy which is based on network coding[2] can effectively avoid the mass redundancy of simple backup strategy. Aly et al.[3] proposed LT Codes[4] based distributed storage algorithm, which can provide persistent data storage solutions with low data redundancy when no geographic or routing information is available. However, LT Codes may cause a serious cliff effect[5].

This paper is organized as follows: First, in section 2, we describe some previous related work in this area. Then in section 3, we introduce the network settings and formulate the problem scenario. In section 4, we provide the details of the proposed BRRCD algorithm. In section 5, we present several simulation results and performance analysis. Finally, in section 6, we make a conclusion of our research.

2 Related Work

In the wireless sensor networks, data storage and data collection strategy is closely related. The storage strategy will affect the collection efficiency directly.

In 2000, in Chinese University of Hong Kong, R.Ahlswed et al.[9] first proposed network coding, so that the multicast of network can always achieve the Shannon maximum flow minimum cut theorem that achieve the maximum transmission capacity of the network. In recent years, distributing storage algorithms based on network coding was proposed, such as Reed-Solomon Codes[6], LDPC Codes[7], and LT Codes[8]. In 2006, Dimakis et al.[10] proposed a distributed fountain codes scheme, but it is difficult to implement in practical applications, because it required the sensor nodes to possess their geographical and routing information. In 2007, YunfengLin et al.[11] introduced random walks to disseminate data from a sensor to subset of sensors, and also proposed a distributed Fountain Codes to achieve data persistence. In 2008, Salah A. Aly et al.[3] proposed Fountain Codes based on distributing storage algorithm, which can provide persistent data storage solutions with low data redundancy when no geographic or routing information is available. In 2009, Hagedorn et al.[12] tried a feedback mechanism to improve the decoding performance. However, this strategy needed to recalculate degree distribution function at the sender. It increased the complexity of computing. In 2012, Jan Wan et al.[13] proposed a prioritized degree distribution strategy, but it needs some global information about the sensor network. In 2012, Wei Zhang et al.[14] proposed a distributed rateless coding method to improve the performance of data collecting. The same year, Wei Zhang et al.[15] proposed a hierarchical topology controlling method to extend network lifetime.

The storage strategy and collection strategy are two main factors affecting the performance of network coding. In view of the two aspects, this paper proposed BRRCD strategy to improve the speed of data distributed and efficiency of data collection, restraining the cliff effect.

3 Network Model and Problem Description

3.1 Network Model

The model of the wireless sensor networks is shown in Figure 1. The number of nodes n is 500, and 310 of 500 is sensor nodes. The size of detected region A is $L * L$. The communication radius is R.

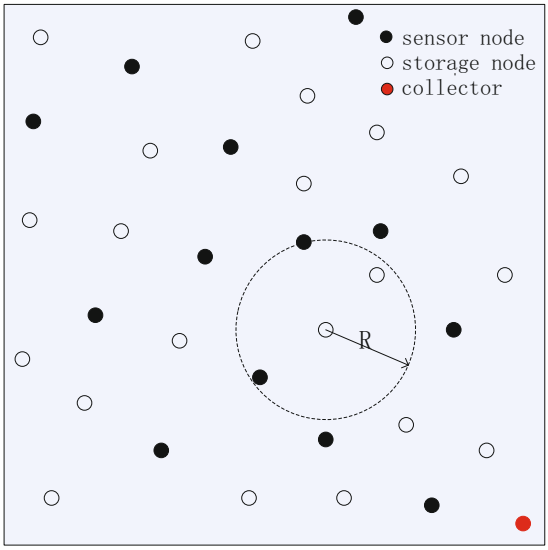


Fig. 1. The model of the wireless sensor networks

Data packet format is shown in Figure 2. Source Nodes ID represents the ID of a node that generated data packet. The $c(x_i)$ represents the hops that the data packet needs to be forwarded. Data represents the data which is generated by the sensor node.

Source Node’ s ID	Counter	Data

Fig. 2. Data packet format

3.2 Problem Description

Each sensor node has the characteristics of limited energy and easy to fault. In order to improve the data persistence and reliability in the network, paper [3] proposed a distributed storage strategy which is based on LT Codes.

This strategy needs no geographic information or routing information, which can provide persistent storage scheme in lower redundancy for data. But it may cause a serious cliff effect [5], when there are only a small number of decoded data at the early period of data collection so that height degree data packets can not be decoded temporarily, causing low decoding rate.

4 BRRCD Algorithm Design

This section introduces LT Codes[4] briefly, and then details BRRCD algorithm.

4.1 LT Codes

Fountain codes[3] is a Rateless code, its idea is to encode k source symbols block randomly with certain probability distribution function, forming a new symbol block, namely the encoded packet. The number of source blocks d employed to generate an encoded block is defined as the coding degree for the fountain codes. Each encoded packet for Fountain codes is obtained by XOR-ing d packets randomly and independently chosen from source packets, where d is drawn from a probability distribution function. The k source blocks can be decoded from any subset of $k(1+\varepsilon)$ encoded blocks and $\varepsilon > 0$ is a small number. The encoding overhead and decoding overhead of fountain codes is very small. LT Codes [4] is a special class of fountain codes, which is designed using Ideal Soliton or Robust Soliton distribution. For k source blocks, the Ideal Soliton distribution Ω_{is} (i) is given by (1)

$$\Omega_{is}(i) = P(d = i) = \begin{cases} \frac{1}{k}, & if\, i = 1 \\ \frac{1}{i(i-1)}, & if\, i = 2, ..., k \end{cases} \tag{1}$$

4.2 BRRCD Algorithm

In this section, we will introduce the strategy BRRCD which is used to collect data packets with rough routing while the degree is increasing gradually.

4.3 BRRCD Algorithm Design

☐ The initialization phase

In the initialization phase, distribute the nodes in the detection area randomly and uniformly. The collector is fixed at the edge of the network where is relatively safe. At this point, the network is not layered. The level of each node and collector in the network is $rank = 0$. When beginning to establish rough route, collector broadcasts a signal. Each node received this signal will firstly check whether its own $rank$ is 0. If $rank = 0$, the $rank$ is set to $rank = rank + 1$; if $rank$ is not 0, it doesnt change. Meanwhile, when node received the signal, it would compare its own $rank$ with the $rank$ of sending node. If its $rank$ is greater than the $rank$ of the sending node, it will record the sending node ID, namely node will record the neighbor node ID which is

up one layer of itself in the network. At this time, the establishment of first layer of the network is completed. The nodes of first layer of the network will broadcast the signal to establish the second, the third layer network and so on, until there is no node whose *rank* is 0 except the collector. Once the rough route is completed, it doesnt need to be updated and maintained. As long as the network connectivity is not destroyed, it must be able to find the route that can reach the collector. During creating the rough route, each node will generate degree d_c (u) randomly which is according to the distribution given by Ideal Soliton Ω_{is}(d). The hierarchy network is shown in Figure 3. K represents the hierarchy of nodes in the network, such as the node 4 belong to the second layer of network, and its next hop neighbor nodes which are up one layer of itself in the network are node 2 and node 3.

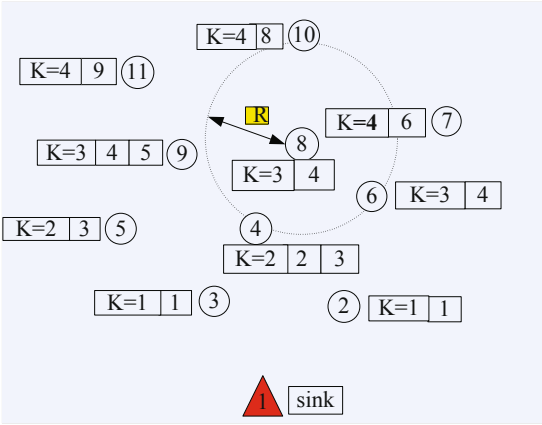


Fig. 3. The hierarchy network

☐ The encoding phase

When node V_i overhearing or receiving a source packet P_i from the neighbor node U_i, if it is the first time V_i receives data packet P_i, it will add the $count(V_i)$ by one. The $count(V_i)$ is to record the total number of different source packets received by node. And then decide whether to accept or reject the data. If count$(V_i) < k$, the node will accept the data with probability $d_c(V_i)/k$. If accepted, it will update the storage as $y^+_{V_i} = y^-_{V_i} \oplus P_i$. But if $count(V_i) = k$, and at the same time the actual encoding degree still remains zero, the node will accept the data with probability one. No matter P_i is accepted or not, node V_i will decrease the hop count of this packet by one as $c(P_i) = c(P_i) - 1$, and puts it into its forward queue.

At the same time, V_i will create a queue Q_i to record node U_i representing that U_i has received the P_i. Then, the node V_i will broadcast a message to tell its neighbors that it has received Pi and the neighbors of V_i will also create a queue Q_i to record node V_i. When V_i is going to forward P_i, it will

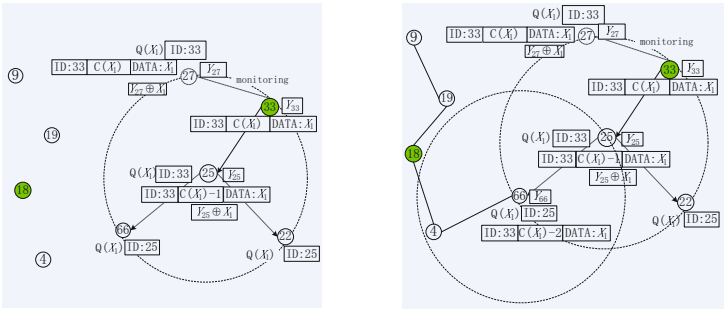


Fig. 4. Data packet forwarding process

choose one of its neighbors that is not in the queue Q_i as the forwarding destination.

If V_i has received packet P_i before, the node does not determine whether to accept or not, but directly decreases the hop count by one as $c(P_i) = c(P_i)-1$, and puts it into its forward queue.

The node V_i will check if the hop count of this packet is zero. If it is not zero, the node will continue sending this data to neighbors and otherwise, it will discard this packet. Figure 4(a) and Figure 4(b) represented the data packet forwarding and encoding process.

Figure 4(a) represented that 33^{th} node generated a source data X_1 and forwarded X_1 to the 25^{th} node, then decreasing the hop count by one as $c(X_1) = c(X_1) - 1$. The 25^{th} node would decide whether to accept or reject the data. If excepted, it would update the storage unit Y_{25} as $Y_{25} \oplus X_1$. The same time, the 27^{th} node overheard the data packet X_1 which was send by 33^{th} node. Also the 27^{th} node will also decide whether to accept or reject the data. Then it would create a queue $Q(X_1)$ to record the 33^{th} node representing the 33^{th} node had received X_1. The 25^{th} node will broadcast information to inform the 66^{th} and 22^{th} nodes that it had received X_1. The 66^{th} and 22^{th} nodes are 25^{th} nodes neighbors. Then the two nodes create a queue $Q(X_1)$ to record the 25^{th} node.

Figure 4(b) represented that 25^{th} node selected the 66^{th} node as the forwarding destination which is one of its next hop neighbor nodes and not in $Q(X_1)$, then decreasing the hop count by one as $c(X_1) = c(X_1) - 1$. Until the $c(X_1)$ is zero, then node will discard X_1. Forwarding data packets with overhearing and recording mechanism can effectively reduce the number of times of invalid forwarding data packets. It can enhance the speed of distributing data packets and persistence of data packets in the network. This strategy saved largely overhead of communication.

☐ The storage phase

When a node records ID of all sources, it means that all their packets have visited the node at least once. The node finishes its encoding phase and would

not update the storage. When all nodes satisfied these conditions, network coding phase is completed, and each node possesses an encoded data packet in its buffer.

- The collection phase
  After the end of encoding phase, collector gathered data depending on increasing the packet-degree gradually. For instance, Firstly collector sent a query message to gather the data packets whose packet-degree is 1. This message would be forwarded to the higher-level of network indistinguishably. When node received query message, it would determine whether the degree of encoding packet is 1. If so, this node would prior to select the forwarding node which had any more next hop neighbors as the forwarding destination. Forwarding those data packets with this strategy, until collector obtained all of data packets whose degree is 1 in the network. Then using the same method, collect data packets whose packet degree is 2, and so on, until recover all of the source data.

5 Experimental Results and Performance Analysis

The simulation experiment of this paper is mainly concentrated on the random distribution of nodes in the network, sensor nodes generating data, nodes forwarding and encoding data, and final a collector to gather and decode data. We use java program to simulate the various parts of the experiment.

In this experiment, 500 nodes are deployed randomly within area of 70 * 70. There are 310 sensor nodes. The communication radius is 14. The average number of neighbors of the node is 50. We call it intensive network. Assume that the link quality is good and the phenomenon of packet loss does not occur. If there is any isolated node, redeploy the network. Nodes generate degree randomly according to the ideal soliton distribution function. The experimental result is shown in Figure 5.

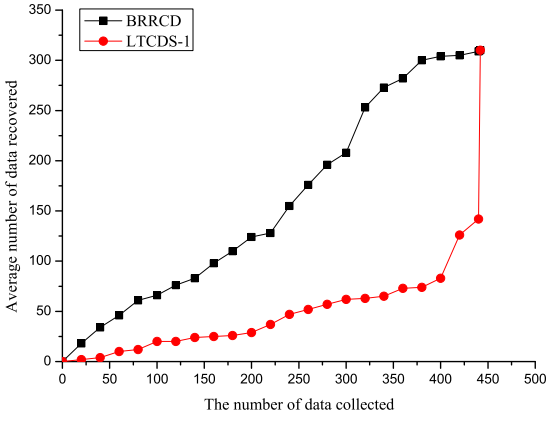


Fig. 5. n=500,k=310,r=14

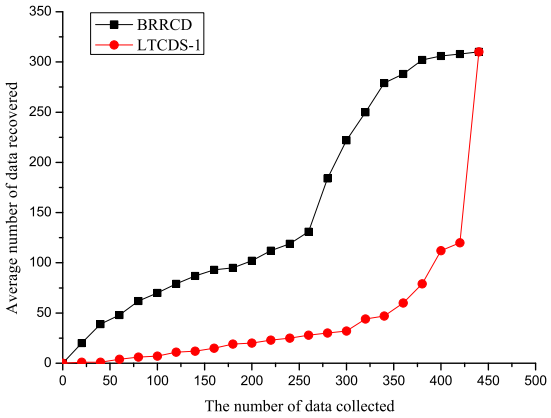


Fig. 6. n=500,k=510,r=8

Figure 5 showed the comparison of decoding performance between BRRCD algorithm and LTCDS-1 algorithm. X-axis represents the average number of encoded data which is gathered by collector while Y-axis represents the average number of original data packets which have been recovered from encoded data. As can be seen from Figure 5, the strategies BRRCD and LTCDS-1 both need about 450 encoded packets to recovery all source packets. BRRCD has achieved linear growth of recovering data approximately. When collector had gathered 425 encoded packets, BRRCD recovered almost all of the source packets. However, LTCDS-1 only recovered 125 source packets. 185 packets still can not be recovered. Obviously, the efficiency of data collection of BRRCD is significantly higher than LTCDS-1. The reason is that BRRCD strategy utilizes rough route to collect data packets depending on increasing packet degree gradually. It overcomes low decoding rate in the early decoding phase.

For examining the performance of BRRCD strategy in the sparse network, we set the communication radius to 8 and other network parameters are not changed. So the average number of neighbors of the node is 20. We call it sparse network. The experimental result is shown in Figure 6. In a sparse network, the strategies BRRCD and LTCDS-1 both need about 430 encoded packets to recovery all source packets. In detail, when collector had gathered 260 encoded packets, BRRCD recovered 130 source packets. However, LTCDS-1 only recovered 28 source packets. So the data collection performance of BRRCD algorithm is also better than LTCDS-1. BRRCD algorithm restrains the 'cliff effect' in the extent.

6 Conclusion

In wireless sensor networks, data distribution and collection strategies are very important to improve the persistence and reliability of data. This paper proposed

BRRCD algorithm which utilizes the overhearing and feedback mechanism in data encoding phase. It is effectively reducing the number of hops which is needed for packets to traverse all of the nodes in the network. BRRCD algorithm largely saves communication cost, while it is improving the speed of distributing data, making data distribution more uniform and enhancing the persistence of data in the network. This strategy effectively restrains the cliff effect of LTCDS-1 Codes. In future work, we will consider more practical wireless sensor network scenarios, such as massive nodes are suddenly failure or death, and how to utilize rough route strategy to ensure the data collection efficiency. And we will also think about data in the adjacent sensor nodes is similar. How to do effective information extraction and information fusion to reduce the data similarity is very important. It can improve the efficiency of collecting data.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (No. 61100193), the Zhejiang Provincial Natural Science Foundation of China (No.Y1090940,Y1090328), the National Key Technology Research and Development Program of the Ministry of Science and Technology of China (No.2012BAH24B04), the Important Science & Technology Specific Projects of Zhejiang Province of China (2010C13022), the Startup Foundation of Hangzhou Dianzi University (No.KYS055608109).

References

1. Yan, W., Fahmy, S., Shroff, N.B.: On the Construction of a Maximum-Lifetime Data Gathering Tree in Sensor Networks: NP-Completeness and Approximation Algorith. In: INFOCOM, pp. 356–360 (2008)
2. Albano, M., Gao, J.: In-network Coding for Resilient Sensor Data Storage and Efficient Data Mule Collection. In: Scheideler, C. (ed.) ALGOSENSORS 2010. LNCS, vol. 6451, pp. 105–117. Springer, Heidelberg (2010)
3. Aly, S.A., Zhenning, K., Soljanin, E.: Fountain Codes Based Distributed Storage Algorithms for Large-Scale Wireless Sensor Networks. In: International Conference on Information Processing in Sensor Networks, IPSN 2008, pp. 171–182 (2008)
4. Luby, M.: LT Codes. C. in Processing. In: IEEE Symp. Foundations of Computer Science (FOCS 2002), pp. 271–282 (2002)
5. Talari, A., Shahrasbi, B., Rahnavard, N.: Efficient symbol sorting for high intermediate recovery rate of LT codes. In: 2010 IEEE International Symposium on Information Theory Proceedings (ISIT), pp. 2443–2447 (2010)
6. Plank, J.S., Xu, L.: Optimizing Cauchy Reed-Solomon Codes for Fault-Tolerant Network Storage Applications. In: Fifth IEEE International Symposium on Network Computing and Applications, NCA 2006, pp. 173–180 (2006)
7. James, S.P.: A Practical Analysis of Low-Density Parity-Check Erasure Codes for Wide-Area Storage Applications. In: International Conference on Dependable Systems and Networks, pp. 115–125 (2004)
8. Puducheri, S., Kliewer, J., Fuja, T.E.: Distributed LT Codes. In: 2006 IEEE International Symposium on Information Theory, pp. 987–991 (2006)
9. Ahlswede, R., Cai, N., Li, S.Y.R.: Network information flow. IEEE Transactions on Theory 46(4), 1204–1216 (2000)

10. Dimakis, A.G., Prabhakaran, V.: Ramchandran, K.: Distributed Fountain Codes for Networked Storage. In: Proceedings of 2006 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP (2006)
11. Lin, Y., Liang, B., Li, B.: Data Persistence in Large-Scale Sensor Networks with Decentralized Fountain Codes. In: 26th IEEE International Conference on Computer Communications, INFOCOM 2007, pp. 1658–1666. IEEE (2007)
12. Hagedorn, A., Agarwal, S., Starobinski, D., et al.: Rateless Coding with Feedback. In: IEEE INFOCOM 2009, pp. 1791–1799 (2009)
13. Wan, J., Xiong, N., Zhang, W., Zhang, Q., Wan, Z.: Prioritized Degree Distribution in Wireless Sensor Networks with a Network Coded Data Collection Method. SENSORS 12(12), 17128–17154 (2012)
14. Zhang, W., He, L., Feng, J., Xu, X., Wan, J.: Distributed Rateless Coding Method for Wireless Sensor Networks Based on Data Oriented Exchanging Strategy. Journal of Computational Information Systems 8(4), 1425–1432 (2012)
15. Zhang, W., Xiong, N., Yang, L.T., Jia, G., Zhang, J.: BCHED – Energy Balanced Sub-Round Local Topology Management for Wireless Sensor Network. Journal of Internet Technology 13(3), 385–394 (2012)

Event Matching Algorithm to Collaborate Sensor Network and the Cloud through a Middleware Service*

Mohammad Hasmat Ullah[1], Sung-Soon Park[2], and Gyeong Hun Kim[3]

[1,2] Department of Computer Science and Engineering, Anyang University, South Korea
[3] Gluesys Co., Ltd. South Korea
{raju,sspark}@anyang.ac.kr, kgh@gluesys.com

Abstract. Cloud computing and virtual communities have been changing the face of communication. Cloud computing covers almost every services and provides most of data services. On the other hand, wireless sensor networks (WSN) gained attention for their potential supports such as environment monitoring, bio-median acknowledgment, healthcare monitoring, industrial automation etc. But this sensor network is not available to community groups or cloud environment for general purpose research or utilization yet. If we can reduce the gap between real and virtual world by adding this WSN produced data to cloud environment and virtual communities to alert the general researchers about environmental situations, it can gain a remarkable attention all over and we can be benefited in all our ability. To do this we need to make collaboration between sensor networks and the cloud environment. Here we have proposed a pub/sub based middleware service for WSN and Cloud platform collaboration which will provide resource, service, and storage with sensor driven data to the community. We proposed a content based event matching algorithm to analyze subscriptions and match proper content easily to convey WSN driven data to subscribers. The validation shows this content matching algorithm works efficiently compared with previously proposed algorithms to integrate WSN to cloud environment to lower the gap between real and virtual world.

Keywords: Event matching, middleware service, WSN, cloud computing.

1 Introduction

Wireless sensor networks have emerged as an exciting computing platform which couples the digital world with physical environment. In vast areas like weather monitoring and forecasting, environmental monitoring, industrial automation WSN have been gaining importance and popularity day by day. They allow the interaction between users and physical environment. WSN contains sensor devices with limited sensing capability, low processing power, and poor communication power.

* This research is supported by WBS (World Best S/W) Development Project, Grants No. 10040957, funded by Ministry of Knowledge Economy 2011.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 118–127, 2013.

On the other hand Cloud computing provides dynamically scalable resources and reliable services delivered through next generation data centers built on virtualization technologies. Cloud computing provides access to applications and data from anywhere in a form of apps. The applications are hosted as "Software as a Service". Only cloud computing can provide unlimited resource, computing power, bandwidth, storage, dedicated servers to access from anywhere anytime to use application like software. If we can utilize both powerful platforms like cloud and WSN together we may gain in both technically and financially.

The rationales behind sensor cloud integration are: super computer may provide resource and power to process sensor data but it is not easily available for general use and needs much overhead. Cloud computing can analyze, process and store the vast amount of data collected by sensors and these sensors can easily be shared by applications and users. Not only cloud provides powerful computation but also serves with huge amount of storage to store processed sensor data for further use.

We propose a middleware for collaborating Cloud and sensor network which will not be topic based rather event based. It will monitor the subscriptions through cloud for sensor driven data service and will receive sensor driven data, will encapsulate as event and will provide to appropriate subscribers. This middleware will deliver information to the subscribers who has subscribed for the sensor driven data.

Our proposed middleware will simplify the integration of sensor network with cloud based community centric applications. The middleware provides an efficient event matching algorithm to provide appropriate sensor driven data to appropriate users.

In Section 2, we review the previous work in this field. Section 3 illustrates the content based middleware and describes our system overview, Section 4 presents our proposed algorithm, Section 5 provides experimental methodology and experimental evaluation of content based event matching algorithm for sensor cloud middleware, and Section 6 states conclusion of our work.

2 Related Work

Not much effort has been taken to address the issue of integrating sensor network to cloud computing. SGIM [4] addresses the opportunity and challenges for sensor-cloud framework only for analyzing healthcare sensor data for range predicate case only. Sensor-Grid [5] architecture is already proposed but grid computing is different than cloud computing [6] and setting up the infrastructure is not easy. Grid focuses on HPC related applications whether cloud focuses on general purpose applications.

Our proposed middleware makes collaboration between sensor networks and cloud which contains content based Pub/Sub model to deliver appropriate data to subscribers. Pub/Sub system encapsulates sensor data into events and provides the service of event publications and subscriptions for asynchronous data exchange. Most notable Pub/Sub systems implemented in recent years are:

The MQTT-S [7] is a topic-based pub-sub protocol that hides the topology of the sensor network and allows data to be delivered based on interests rather than

individual device addresses. It allows a transparent data exchange between WSNs and traditional networks and even between different WSNs. In [8] Mires is a publish/subscribe architecture for WSNs. Basically sensors only publish readings if the user has subscribed to the specific sensor reading. Subscriptions are issued from the sink node which then receives all publications. Subscriptions are made based on the content of the desired messages in DV/DRP [9]. Subscriptions are flooding over the network and it publishes data if there is some subscription for it.

Several event matching algorithms are proposed to deliver published sensor data or events to subscribers. In Sequential and sub-order [10] algorithm, according to each predicate in event, searching space is gradually reduced by deleting subscriptions which are not satisfied. The second algorithm, sub-order, reduces the expected number of predicate evaluations by analyzing the expected cost differences when subscriptions are evaluated in different orders. But it is difficult to make chain sometimes, for instance in range predicate case and here every two predicates, if they are same, and share a chain. So it needs to maintain a complete graph and bring so much overload when inserting and deleting subscriptions.

3 The Middleware Service

3.1 Content Based Middleware

Our current environmental forecasting system which provides pre-processed data may always not be enough. Still we doesn't have any forecasting system for earth quack but we may put some effort to alert others by providing real time auto generated information when sensor gets such information about natural calamities just started to take place by passing sensor driven data to cloud environment through some collaborating middleware to share with the community. On the other hand researchers who are trying to solve some complex problem and need data storage, computational capability, security, and they need it all provided at the same time to process vast amount of real time data. For example, this team is working on an unusual environmental situation. They plotted sensors on some specific region to monitor the magnitude continuously and to use this data for large multi-scale simulations to track the natural calamities along with provide auto generated forecast to the end users who has subscribed to know the forecast. This may require computational resources and a platform for sharing data and results that are not immediately available to the team. Traditional HPC (High Performance Computing) approach like Sensor-Grid model can be used in this case, but setting up the infrastructure as mentioned above is not easy in this environment. Amazon EC2 like Cloud data center shows that many copies in a data center will work. But they did not address the issue of integrating sensor network with Cloud applications and thus have no infrastructure to support this scenario. Here, the subscribers need to register their interests to get various environmental states (magnitude, temperature of ionosphere, electromagnetic field etc.) from sensors for large scale parallel analysis and to share this information with each other to find useful solution of the problem. So the sensor data needs to aggregate, process and disseminate based on subscriptions.

3.2 System Overview

In our proposed system we have a pub/sub based middleware to make collaboration between cloud and WSN to provide appropriate data to appropriate subscribers. WSN generates real-time data and needs to be processed at the same time. Our proposed middleware connects to such WSN and receives real-time data, processes them and prepares those data as events rather than topic. The sensor data comes in many forms like raw data that must be captured, filtered and analyzed real-time and sometime is should be stored and cached for further use. It also has registry, analyzer and disseminator. Subscribers can subscribe through cloud API for sensor data.

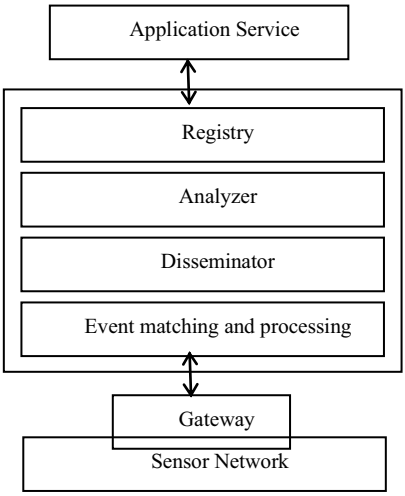


Fig. 1. Simple architecture of middleware

There may be two kind of subscription, general purpose for end users or community based users to get processed data like forecast about earthquake or natural calamities and another may be for sensor's data encapsulated as event for further research. Subscribers who have interest make subscription through cloud application; this subscription will be stored and categorized. The Pub/Sub middleware receives sensor driven data from the gateway between WSN and the middleware, event matching and monitoring section encapsulates these data as event and passes to the analyzer. Analyzer analyzes subscription type and disseminator provides corresponding data to subscribed users by matching the registry through cloud API using event matching algorithm for its further use. The cloud environment may manage these data, process them and may also keep to the repository for further utilization as needed. General user will be able to get user friendly output of these complex data by matching its predicates and by normalizing it.

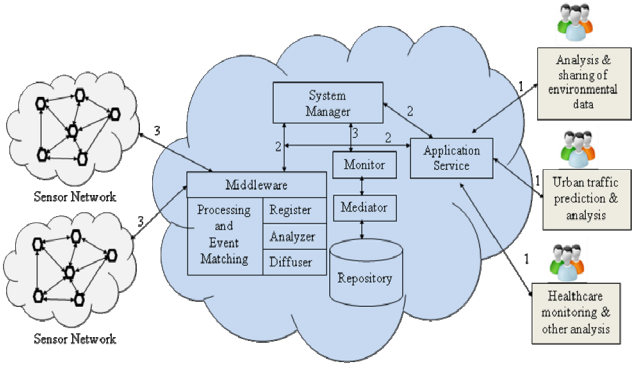


Fig. 2. Middleware service collaborating WSN and Cloud

Above figure shows the overview of proposed system. Our proposed middleware service is integrated with cloud platform joining the WSN with cloud. Cloud service providing application for users to subscribe to their interest as needed. The proposed event matching algorithm will provide appropriate data efficiently to the subscribers.

4 Event Matching Algorithm

We need an efficient event matching algorithm for our system to deliver published events to appropriate subscribers. Our target is Cloud-based environmental data monitoring and analysis system where researchers express their interests into attributes and general end users who just needs easy to understand outputs. First we implemented to support range predicates and to cover multi range data only, then we extended the algorithm to support overlapping predicates also.

4.1 Event Matching

In our system, a subscription S is expressed by a pair (ID , C_i, P_i) where ID is the subscriber's ID, C is subscription category and P is a set of predicates specifying subscriber's interest.

Here is an example of a subscription and an event in the system Subscription: S [magnitude, 7(+), ionosphere temperature, 300K(+)] which contains two predicates that are joined together to specify a discrete value predicate, here magnitude 7(+) represents 7 and more alternately ionosphere temperature 300K(+) indicates from 300K to max i.e. P_1 = magnitude >= 7 and P_2 = ionosphere temperature >= 300K. We also can express it as 6.9 < magnitude < 8 and 299K < temperature < 500K. Let event e: [magnitude = 7.6, ionosphere temperature = 350K].

An event satisfies a subscription only if it satisfies all predicates in the subscription. Here, the event [magnitude = 7.6, ionosphere temperature = 350K, 375K] satisfies the subscription S as our proposed method supports discrete predicate values also. So the matching problem is: Given an event e and a set of predicates in

subscription set S. We need to find all subscriptions in set S that are satisfied by e. Our middleware supports various expressions of predicates. First, "(data >= LV or vice versa)" [here LV = lower value and UV = upper value] is used when consumers want to know normal patterns of sensed data. Second, "(LV > data || UV < data)" is used when consumers need to receive unusual states of the situation.

4.2 Proposed Method

Here we describe the Category wise Matching Algorithm (CMA). This algorithm operates in three stages. In first stage it pre-processes subscriptions by categorizing them by the predicates corresponding to the relevant properties of events. The basic categorizing idea from statistics is employed to decide the number of category. In the second stage matching subscriptions or predicates are derived sequentially. All predicates stored in the system are associated with a unique ID. Similarly subscriptions are identified with subscription id. And in final stage it will store the sensor driven data to knowledgebase for future analysis.

Here suppose S is a set of subscriptions, $S = \{s_1, s_2, \ldots s_{n-1}, s_n\}$ where n is total no. of subscriptions and P is a set of predicates in S, $P = \{p_1, p_2, \ldots p_{m-1}, p_m\}$ where m is the total no. of predicates in a subscription. In our system, we have two predicates in a subscription (i.e. data > LV and data < UV) and these two predicates is used to categories the subscriptions. We define a set S´ that contains all the subscriptions of S sorted by LV value in ascending order. Then we define a categorizing sequence (mC_1, $mC_2, \ldots mC_g$). The categorizing space, denoted by SP (S´, c), is defined as the set containing all such category sequences over S´ and c. Now each $mC_{i=1\ldots c} \in SP$ (S´, c) contains k = n/c no. of subscriptions, so category index is created for each $cI_i \in mC_{i=1\ldots c.}$ Here this categorizing sequence is called almost balanced categorizing sequence since every category contains same no. of subscriptions except the last one which may or may not contain the same no. of subscriptions. It depends on the value of c and n.

When categorizing of subscriptions is done in the above way, first predicate of an event is compared with category index $cI_1 \in mC_1$ and if any match found then second predicate is compared with category indexes $hI_i \in MC_{i=1\ldots h}$. In this way all categories are found that matches with event data. Finally sequential matching is done in the selected categories to find the subscriptions that are satisfied by all predicates in the event.

5 Evaluation

Our experimental methodology and simulation results are presented in this section. We may compare our proposed method with sequential sub-order, forwarding and naïve algorithms. Naive, a baseline algorithm, evaluates the subscriptions independently. Specifically, it evaluates the subscriptions one by one and, for each subscription, evaluates its predicates until either one of them becomes false or all of them are evaluated.

5.1 Experimental Methodology

To evaluate pub/sub system is the main challenge as for lack of real-world application data. However, previous work shows that in most applications events and subscriptions follow uniform or Zipf distributions. We did our experiments with both of these distributions along with exponential distribution. We use subscription evaluation cost like as the metric of interest which is the average number of predicates that the system evaluates in matching an event to the subscriptions. We believe that it can well reflect the efficiency of the evaluation process.

5.2 Experimental Results

We have compared Naïve, sequential and sub-order algorithm with our CMA using uniform distribution. The experiment results of evaluation are given below:

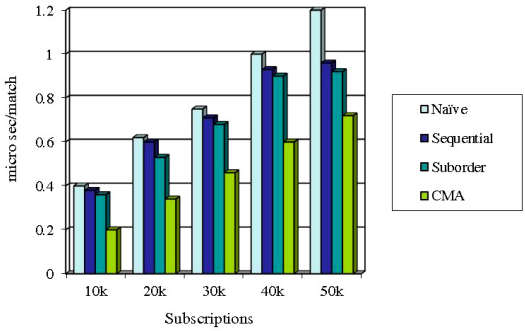


Fig. 3. Matching time vs. number of subscriptions

From first comparison, we can observe that CMA performs better than all other algorithms. For example, with 10K subscriptions and 5000 events, the naïve, sequential, sub-order and CMA evaluate predicates in 0.4, 0.38, 0.36, 0.2 micro sec respectively. Thus CMA reduces evaluation cost by 50%, 42%, and38%as compared to naïve, sequential, sub-order respectively.

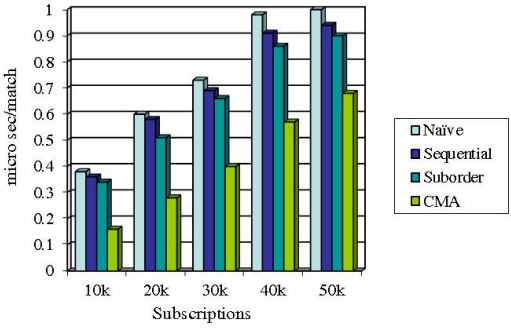


Fig. 4. Matching time vs. number of subscriptions (Zipf Distribution)

Again we repeat the experiments with the same parameter settings except that the distribution now follows Zipf rather than the uniform distribution. The experiment results exhibits exactly the same trend as in first comparison.

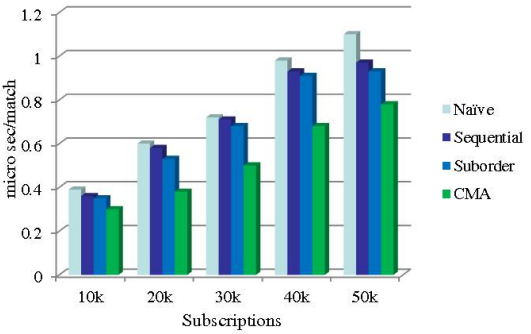


Fig. 5. Matching time vs. number of subscriptions (Exponential Distribution)

Figure 5 shows similar performance if we use same parameter settings with exponential distribution (p = 2). The experiment result exhibits similar trend as in above comparisons except this time performance of each algorithm is little closer to each other.

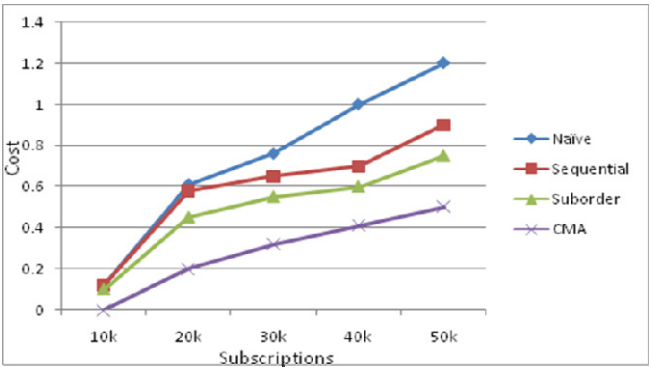


Fig. 6. Evaluation cost for multi-range predicates

From figure 6 we can observe that for multiple ranges of predicates our algorithm performs much better than any other. For example beginning from 10k subscriptions and 5000 events Naive, Sequential and sub-order event matching performed 35% ~ 55% poorer than CMA. The cost is evaluated in micro seconds.

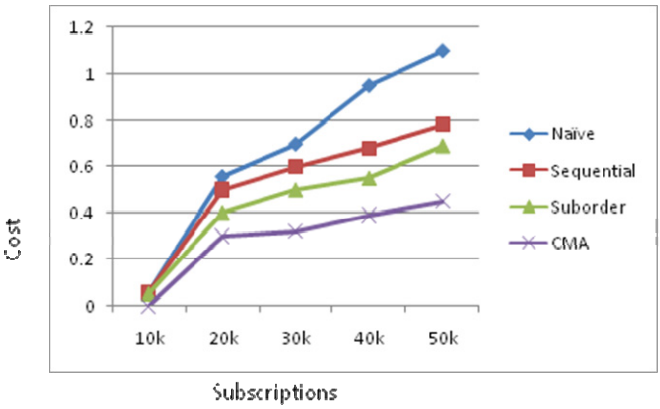


Fig. 7. Evaluation cost for overlapping predicates

Figure 7 shows the comparison result for the overlapping predicates. As the subscription increases CMA shows better and better performance than others. So it will outperform if the subscriptions are larger.

Above experiments clearly shows that our CMA algorithm performs better (in case of uniform, Zipf and exponential distributions) than existing ones in terms of efficiency and scalability.

6 Conclusion

In this paper, we have proposed a middleware service for collaborating sensor network to cloud environment for utilizing the ever-expanding sensor data for various next generation community based sensing applications over the Cloud. The computational tools needed to launch this exploration is more appropriately built from the data center "cloud" computing model than the traditional HPC approaches or Grid approaches. We proposed a middleware to enable this by content-based pub-sub model. Also to deliver published sensor data or events to appropriate users of Cloud applications, we propose an efficient and scalable event matching algorithm. We evaluated its performance and compared with existing algorithms in a cloud based environment analysis scenario. In future, we will study more to make the middleware more efficient to distribute sensor driven data to appropriate subscribers and will try to simplify the communication overhead between WSN and cloud environment. We are also considering compression of sensor driven data after processing to better utilize repository along with finding efficient resource negotiation algorithm for efficient processing. We will make our system generic rather than specific application oriented considering easily accessible in a cost effective manner.

References

1. Buyya, R., Yeo, C.S., et al.: Market Oriented Cloud Computing: Vision, Hype and Reality for Delivering IT Services as Computing Utilities. In: Proc. of 10th IEEE Conference on HPCC, pp. 5–13 (2008)
2. Khedo, K.K., Subramanian, R.K.: A Service-Oriented Component-Based Middleware Architecture for Wireless Sensor Networks. International Journal of Computer Science and Network Security 9(3), 174–182 (2009)
3. Weissp, A.: Computing in the Clouds. Networker Magazine 11(4), 16–25 (2007)
4. Hassan, M.M., Song, B., Huh, E.-N.: A framework of sensor-cloud integration opportunities and challenges. In: ICUIMC 2009 Proceedings, pp. 618–626. ACM (2009)
5. Lim, H.B., Teo, Y.M., Mukherjee, P., Lam, V.T., et al.: Sensor Grid: Integration of Wireless Sensor Networks and the Grid. In: Proc. of the IEEE Conf. on Local Computer Networks, pp. 91–98 (2005)
6. Harris, D.: The Grid Cloud Connection: Compare and Contrast, `http://www.on-demandenterprise.com/features/The_Grid-Cloud_Connection_Pt_I_Compare_and_Contrast_30634819.html`
7. Hunkeler, U., Truong, H.L., Stanford-Clark, A.: MQTT-S – A Publish/Subscribe Protocol For Wireless Sensor Networks. In: IEEE Conference on COMSWARE, pp. 791–798 (2008)
8. Souto, E., et al.: Mires: A publish/subscribe middleware for sensor networks. ACM Personal and Ubiquitous Computing 10, 37–44 (2005)
9. Hall, C.P., Carzaniga, A., Rose, J., Wolf, A.L.: A content-based networking protocol for sensor networks. Department of Computer Science, University of Colorado,Technical Report (2004)
10. Liu, Z., et al.: Scalable Event Matching for Overlapping Subscriptions in Pub/Sub Systems. In: Proc. of DEBS 2007, pp. 250–261. ACM Press (2007)
11. Gaynor, M., et al.: Integrating wireless sensor networks with the grid. IEEE Internet Computing, 32–39 (2004)
12. Eugster, P.T.: The many faces of Publish/Subscribe. ACM Computing Surveys 35(2), 114–131 (2003)

An Efficient Map-Matching Mechanism for Emergency Scheduling and Commanding

Yaguang Li[1,2], Kuien Liu[1], Jiajie Xu[1], and Fengcheng He[1,2]

[1] Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
{yaguang,kuien,jiajie,fengcheng}@nfs.iscas.ac.cn
[2] University of Chinese Academy of Sciences, Beijing 100049, China

Abstract. Efficient vehicle tracking and trajectory analyzing are important to emergency scheduling and commanding as they are essential for assessing and understanding the current situation. One of the fundamental techniques is map matching which aligns the trajectory points of moving objects to the underlying traffic network. In this paper, we propose an efficient map matching algorithm called EM^3 to meet the requirement of high efficiency and accuracy posed by emergency management. Instead of matching every single GPS point, the algorithm concentrates on those close to intersections and infers the matching results of intermediated ones, which makes the algorithm quite efficient and robust to edge simplification. To provide accurate matching results in ambiguous situations, e.g., road intersections and parallel paths, we further propose EM^3*, which is based on the multi-hypothesis technique with novel candidate generation and management methods. The results of experiments performed on real datasets demonstrate that EM^3* is efficient while maintaining the high accuracy.

Keywords: Efficient Map-matching, Emergency Management.

1 Introduction

As an efficient way to minimize the disaster losses, emergency management has received increasing attentions in recent years. In order to do scheduling and commanding in many situations like emergency evacuation and emergency materials distribution, it is important for government authorities to gain an overview of the current condition, especially the traffic network. Undertaking this non-trivial task primarily relies on applications of the Intelligent Transport Systems, e.g., traffic flow analysis [1], accident detection[2], route planning [3], evacuation scheduling [4]. All of these applications require analysis of large amounts of trajectory data. An essential pre-processing step that matches the trajectories to the road network is needed. This technique is commonly referred as map-matching.

However, existing map-matching techniques cannot fully satisfy the requirements of emergency management. Some local/increment map-matching methods [5,6,7] are fast but generate inconsistent matching results since they ignore the correlation between subsequent points and topological information of the

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 128–139, 2013.

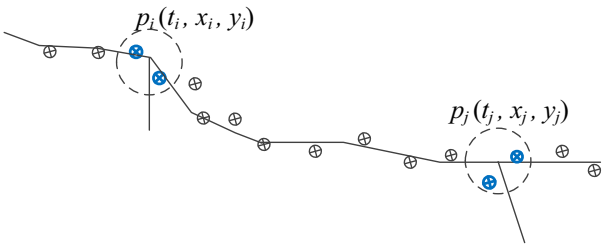


Fig. 1. Matching points close to intersections

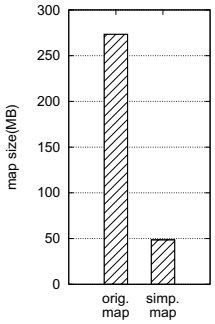


Fig. 2. Map Size

road network. Meanwhile, more advanced map-matching methods, e.g., global approach [8,9] and statistical approach [10,11], manage to be quite accurate, but suffer from high computational complexity. On the other hand, the server-based map matching methods are not applicable to conditions when the communication with server is not available which frequently happens in time of emergency. Thus, map matching has to be performed locally in mobile terminals with limited (energy, storage) resources.

To meet these requirements, we propose an efficient map-matching algorithm called EM^3. Instead of matching every single GPS point, we concentrate on those close to intersections and avoid the computation of map-matching on intermediate GPS points. In the circumstances of constrained road network, if a moving object passes by the start vertex of edge e at t_i, and the end vertex of e at a reasonable time t_j, the object is then supposed to be on e at the time interval between t_i and t_j. Figure 1 illustrates an example of such case, i.e., we just have to match these thick blue points close to intersections. Benefiting from this feature, geometric shape information of the edge can be eliminated through simplification. Consequently, storage and memory size required by the algorithm becomes much smaller, which makes it more feasible to efficiently perform map-matching in mobile terminals when needed. Figure 2 shows the size comparison of the original and the simplified map.

While cutting the cost of computing on intermediate GPS points makes map-matching more efficient, nevertheless it is not at no cost. The uncertainty for determining the real route of the moving object also arises due to less availability of trajectory information. The challenge of this work is to keep providing accurate matching results in ambiguous situations, e.g., road intersections and parallel paths. To handle this challenge, we further improve the algorithm using the multi-hypothesis technique with novel candidate generation methods. Meanwhile, several efficient approaches for candidate management are developed to further improve the performance of EM^3*.

To summarize, this paper makes the following contributions:

- We propose two efficient map matching algorithms. They avoid large parts of computation by only performing map-matching on GPS points close to intersections and inferring the remaining ones.
- We provide a method to greatly reduced the map size while maintaining roughly the same matching accuracy. As EM^3* mainly bases on positions of intersections and road connectivity, it manages to be quite accurate even when geometric information of the edge is not available.
- We conduct experimental study on real dataset and demonstrate the efficiency and accuracy of the proposed algorithms.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 gives the problem statement. The proposed algorithm is described in Section 4. The results of experiments and the analysis are presented in Section 5. Finally, conclusions and future work are summarized.

2 Related Work

In this section, we present a brief overview of previous studies on map-matching and the multi-hypothesis technique.

2.1 Map-Matching Techniques

Map-matching is commonly referred as the process of aligning a sequence of observed positions with the road network. Approaches for map-matching algorithms can be categorized into three groups: local/incremental method [6,7,12], global method [8,9,13], and statistical method [10,11]. The local/incremental method tries to find local match of geometries, and performs matching based on the previous matching result of a point. The global method aims to find a global optimal match for the entire trajectory and a path in the road network. The algorithm in [13] is based on Frechét distance and its variant. [8] proposes an algorithm that takes temporal information into consideration and manages to get a high accuracy for low-sampling-rate GPS trajectories. Statistical method is also widely used. A method based on the Bayesian classifier is presented in [11]. [10] proposes an algorithm that takes advantage of the rich information extracted from the historical trajectories to infer the real routes.

Many advanced techniques are also integrated with map-matching [14]. [15] proposes a fuzzy logical based method that is able to take noisy, imprecise input and yield crisp output. [9] proposes a method based on the Hidden Markov Model which can deal with the inaccuracies of the location measurement systems.

Most existing map-matching algorithms have to perform map-matching on every single trajectory point which is not always necessary. Instead of matching every single GPS point, we concentrate on those close to intersections and infer the results for the remaining points to avoid extra computation.

2.2 Multi-hypothesis Technique

The Multi-Hypothesis Technique(MHT) is a method to track multiple targets under the clutter environment using a likelihood function [16]. To realize a map matching method using the MHT, pseudo-measurements are generated utilizing adjacent roads of GPS position and the MHT is reformulated as a single target problem [17]. The main advantage of multi-hypothesis map-matching over a mono-hypothesis approach is that it maintains all the possible solutions in situations of ambiguity and eventually chooses the one with the highest probability.

3 Problem Statement

We have a standard road network representation in which intersections are represented by vertices, roads are represented by directed edges and the geometric detail of roads are described by polylines.

Definition 1 (Road Network). *The road network G is defined as:*

$$G = (V, E)$$

where V is the set of vertices while E is the set of directed edges. A vertex of G is defined as: $v = (vid, lat, lng)$, where vid is the identifier of v while lat and lng denote the latitude and longitude of v. An edge of G is defined as: $e = (eid, geo, len, v_{from}, v_{to})$, where eid is the identifier of e, len is the length of e, v_{from} and v_{to} represent the start and the end vertices of e, geo is a polyline representing the geometry of e. As mentioned in Section 1, geo is not necessary in proposed algorithms which can greatly reduce the storage and memory consumption.

Definition 2 (Path). *A path P is defined as a sequence of edges:*

$$P = (e_i)_{i=1}^{n} \; n \geq 1 \; \wedge \; \forall 1 \leq i < n \;\; e_{i+1}.start = e_i.end$$

The start vertex and the end vertex of P are defined as:

$$P.start = e_1.start \qquad P.end = e_n.end$$

Definition 3 (Trajectory). *The trajectory of a moving object is defined as a sequence of points with timestamps:*

$$T = (p_i)_{i=1}^{n} = (t_i, lat_i, lng_i)_{i=1}^{n}$$

We use $T[i]$ to represent the ith point of T and $\overrightarrow{p_i, p_{i+1}}$ to represent the ith trajectory segment of T. $\overrightarrow{p_i, p_{i+1}}$ is defined as the directed line segment formed by p_i and p_{i+1}.

Definition 4 (Matching Result). *Given a trajectory* $T = (p_i)_{i=1}^{n}$*, the matching result R is defined as:*

$$R = (p_i, e_i)_{i=1}^{n}$$

where e_i denotes the edge that p_i be matched to.

Definition 5 (Matching Accuracy). *Given the matching result* $R = (p_i, e_i)_{i=1}^n$ *and the ground truth* $R_t = (p_i, \hat{e}_i)_{i=1}^n$*, the accuracy* A *is defined as:*

$$A = \frac{\sum_{i=1}^{n} \theta_i}{n} \qquad \theta_i = \begin{cases} 1 & \text{if } R.e_i = R_t.\hat{e}_i \\ 0 & \text{otherwise} \end{cases}$$

Problem Statement. The **Map-matching** problem can be described as, given a road network G and a trajectory T, how to efficiently find a matching result R, with the goal of maximizing the matching accuracy A.

4 Proposed Algorithms

In this section, we first illustrate the key observations followed by the algorithm description and implementation, then we describe several novel approaches in detail for candidate generation, evaluation and management.

Observation 1. *In most cases, if a moving object passes by both the start and the end intersections of an edge within a reasonable period of time, the object is supposed to be on that edge during the time interval.*

According to Observation 1, we can improve the efficiency of map-matching algorithm by avoiding the computation on intermediate GPS points.

Observation 2. *The representation of vertices is generally more accurate than edges especially when edges are simplified.*

The commonly used representation of vertex and edge in a road network is point and polyline. When the road network is simplified [18,19], the deviations between real edges and polylines can be up to hundreds of meters due to edge simplification or edge measurement error while the representation of vertices tends to be more accurate.

4.1 Map-Matching According to Passed Intersections

Based on above observations, we design a map-matching algorithm called EM^3, which mainly depends on positions of intersections and road connectivity. In particular, this algorithm does not need either the heading information or the instantaneous speed of the vehicle from the dead reckoning device.

One of the key processes in EM^3 is to estimate how likely the moving object passes by a certain vertex. We measure it using the passby probability.

Definition 6 (Passby Probability). *The passby probability is defined as the likelihood that the moving object passes by the vertex* v*.*

$$F_p(T, v) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{\lambda^2}{2\sigma^2}} \qquad \lambda = d(T, v) = \min_{0<i<|T|-1} d(\overrightarrow{p_i, p_{i+1}}, v)$$

Algorithm 1. $\mathbf{EM}^3(G, T)$

1: $(v_c, i_c) \leftarrow \mathbf{init}(G, T, 1)$
2: **while** $i_c \leq n$ **do**
3: $S_c \leftarrow \emptyset,\ S_e \leftarrow \{e | e.start = v_c\}$
4: **for each** $e \in S_e$ **do**
5: $i \leftarrow \mathbf{pass}(T, e.end, i_c)$
6: **if** $i > 0$ **then**
7: $S_c \leftarrow S_c \cup \{(i, e)\}$
8: **if** $S_c \neq \emptyset$ **then**
9: $t \leftarrow \arg\min_{t \in S_c}(t.i)$
10: match $\{p_{i_c+1}, \ldots, p_{t.i}\}$ to edge $t.e$
11: $i_c \leftarrow t.i,\ v_c \leftarrow t.e.end$
12: **else**
13: $(v_c, i'_c) \leftarrow \mathbf{init}(G, T, i_c)$
14: match $\{p_{i_c+1}, \ldots, p_{i'_c}\}$ to nearest edges
15: $i_c \leftarrow i'_c$

where λ is the distance between the trajectory T and the vertex v, i.e., the minimum distance between the segments of T and v. We use a zero-mean normal distribution with a standard deviation of 6.4m for the training dataset.

Algorithm 1 gives an overview of EM^3. It takes two arguments: the road network G, and the trajectory $T = (p_i)_{i=1}^n = (t_i, lat_i, lng_i)_{i=1}^n$. The function $\mathbf{init}(G, T, i)$ takes three arguments: the road network G, the trajectory T and the start index i, to find the first passed vertex v_c that $d(T, v_c)$ is less than a threshold σ_p, e.g., 25m, and the index of corresponding trajectory segment i_c. This can be efficiently calculated by building indices on GPS points and vertices. The function $\mathbf{pass}(T, v, i)$ also takes three arguments: the trajectory T, the vertex v, and start index of trajectory segment i, and returns the index of the nearest trajectory segment from v, or 0 if no valid trajectory segment is found.

First, the algorithm performs the initialization to get the first passed vertex v_c and the corresponding index of the trajectory segment i_c. (line 1). Then the algorithm repeats the following process until the end of trajectory: 1) it retrieves all the candidate edges (line 3); 2) it filters out infeasible edges using passby probability (line 4 - line 7); 3) if valid edges exist, the edge $t.e$ with the smallest index is selected (line 8 - line 9), then the algorithm matches the points between p_{i_c+1} and $p_{t.i}$ to $t.e$ (line 10), otherwise a re-initialization will be made to find the next vertex v_c and corresponding index (line 13). The intermediate points will be matched to their nearest edges (line 14). As illustrated in Figure 1, the algorithm needs only to match the initial and the final points in most cases and is consequently very efficient.

4.2 Use of Multi-hypothesis Technique

The EM^3 algorithm works well in the premise that every passed vertices can be found and the one closest to the moving object is the real passed vertex. However, this assumption may not always hold, e.g., several vertices can be quite close to the moving object at the same time while the real passed vertex might not be close enough to be found. To generate stable matching results in ambiguous situations, e.g., road intersections, we propose another algorithm called EM^3*, which is integrated with the multi-hypothesis technique to maintain all the possible solutions in situations of ambiguity and eventually choose the one with the highest probability.

4.2.1 Candidate Generation

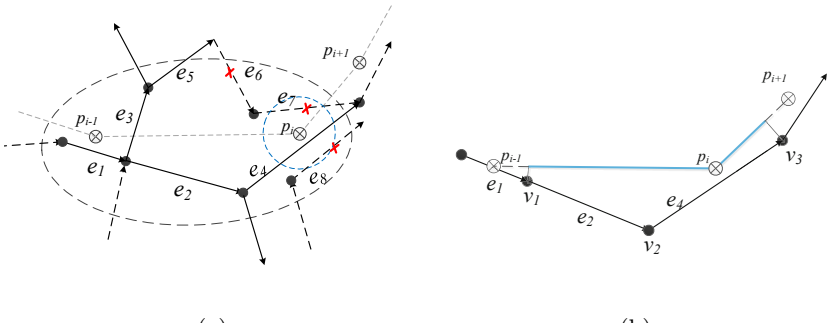
The traditional candidate generation method usually create new candidates at each step, which will result in an exponential computational complexity. To be efficient, we redesigned the rule of candidate generation.

Algorithm 2 shows the pseudo-code of candidate generation process. The algorithm takes three arguments: the road network G, the trajectory $T = (t_i, lat_i, lng_i)_{i=1}^{n}$, the path structure $sp = (v, i, f, ptr)$, and returns set of candidate paths. v is the vertex, i is the index of the trajectory segment that is nearest to the v, f is the score of the path and ptr is the pointer to the parent structure, which is used to get the whole path.

Algorithm 2. generateCands(G, T, sp)

1: $S_{sp} \leftarrow \emptyset$
2: $S_v \leftarrow \{v | d(v, T[sp.i]) + d(v, T[sp.i+1]) \leq 2a\}$
3: $S_P \leftarrow$ **buildPath**(G, S_v, sp)
4: **for each** $P \in S_P$ **do**
5: $\quad i \leftarrow$ **pass**$(T, P.end, sp.i)$
6: $\quad$ **if** $i > 0$ **then**
7: $\quad\quad f \leftarrow sp.f \cdot F_p(T, P.end) \cdot F_t(T, P)$
8: $\quad\quad S_{sp} \leftarrow S_{sp} \cup \{(P.end, i, f, sp\}$
9: **return** S_{sp}

Take Figure 3(a) for example, the algorithm first retrieves all the edges whose start vertices lie within the error ellipse that has a major axis of $2a$ (line 2). e_6 is not a valid edge as its start vertex falls outside the ellipse. e_7, e_8 won't be considered either as they are not connected with valid edges. Then the algorithm builds candidate paths start from $sp.v$ based on road connectivity (line 3). Finally, the algorithm finds these candidate paths whose end vertices are considered passed by the moving object, and calculates their scores using passby probability and vertex transition probability (line 4 - line 8).



(a) (b)

Fig. 3. Examples of candidate generation

Definition 7 (Vertex Transition Probability). *The vertex transition probability is used to measure the the likelihood that the moving object transfers between vertices, and is defined as:*

$$F_t(T, P) = \exp(-\alpha_t \frac{\sum_{e \in P} \text{len}(e)}{\sum_{e \in P} \text{len}(\text{proj}(e, T))})$$

where α_t is the scaling factor, $\text{len}(e)$ returns the length of e, and $\text{proj}(e, T)$ returns the projection of e on T. Figure 3(b) shows the candidate path $\{e_2, e_4\}$, the thick blue line represents the projection of the path on T.

4.2.2 Management of Candidate Paths

The candidate management is mainly used to reduce the candidate size while preserving all possible solutions. It consists of two aspects: pruning and confirmation. Pruning is the process of eliminating infeasible candidate paths while confirmation is the process of confirming a candidate path as the real path.

Pruning happens in the following conditions: 1)The ratio of the score of a candidate to the largest score is lower than a threshold σ_A, which usually happens near road intersections. 2) In the case of parallel paths, all the parallel paths should be merged into a single one.

Confirmation happens in the following conditions: 1) The ratio of the largest score to the next largest one exceeds a threshold σ_B. 2) There exists only one candidate. σ_A and σ_B are two thresholds used to control the candidate size.

4.2.3 Overview of EM^3*

Algorithm 3 outlines the framework of EM^3*. The algorithm takes two arguments: the road network G, the trajectory $T = (t_i, lat_i, lng_i)_{i=1}^n$, and returns the matching result. S_O, S_P are sets of path information structures, i.e., $sp : (v, i, f, ptr)$. Function **getInitialCands**(G, T) is used to get the initial vertices and evaluate them with the passby probability.Function **generateCands**(G, T, sp) is used to get candidate paths with the method proposed in Section 4.2.1.

Algorithm 3. EM^3*(G, T)

```
1: S_O ← ∅; sp_c ← (null, 1, 1, null) //(v, i, f, ptr)
2: while sp_c.i ≤ n do
3:     if S_O = ∅ then
4:         S_O ← getInitialCands(G, T, sp_c.i)
5:     while S_O ≠ ∅ ∧ sp_c.i ≤ n do
6:         sp_c ← first(S_O); S_O ← S_O\{sp_c};
7:         S_P ← generateCands(G, T, sp_c) //Section 4.2.1
8:         S_O ← pruneConfirm(S_O) //Section 4.2.2
9: sp_c ← arg max_{sp∈S_O} (sp.f)
10: R ← getMatchResult(sp_c)
11: return R
```

The algorithm first gets initial paths in line 4, and then get and remove the first item from S_O, i.e., the item with smallest i, denoted as sp_c (line 5). Candidate generation method proposed in Section 4.2.1 are used to create candidate paths (line 7). The pruning and confirmation process are performed in line 8. Finally, the algorithm finds the path with the maximum score in S_O (line 9), and generates the matching result (line 10).

4.3 Theoretical Analysis

Now we will analyze the complexity of the **EM^3*** algorithm. Let n be the length of the trajectory T, l be the maximum number of edges in the error ellipse used in the candidate generation process, and k be the maximum number of candidates.

The time complexity of the function **generateCands**, **pruneConform** and **getMatchResult** are $O(l \log(l))$, $O(k)$ and $O(n)$. In the worst-case, candidate generation might be performed on every single trajectory point, so the EM^3* algorithm has the time complexity of $O(nkl \log(l) + nk + n) = O(nkl \log(l))$. In practice, k is usually quite small, thus the time complexity of EM^3* is close to $O(nl \log(l))$. Furthermore, as indicated by Figure 1, large part of computation is avoid, so the constant factor is actually quite small, this is also confirmed by the experiment.

5 Experiments

In this section, we first present the experimental settings, then we evaluate both the efficiency and the accuracy of proposed algorithms.

5.1 Dataset and Experimental Setup

In our experiment, we use the road network and trajectory data provided by ACM SIGSPATIAL Cup 2012 [20], which is a GIS-focused algorithm competition hosted by ACM SIGSPATIAL. The road network graph contains 535,452 vertices and 1,283,540 road segments. Ground truth files are included in trajectory data file,

which are used in results verification. These algorithms have been implemented with C++ on Visual Studio express platform, the results of the experiments are taken on a computer with Intel Core2 Dual CPU T8100 2.1 GHz and 3 GB RAM.

Baseline Algorithms. We compare proposed algorithms with the incremental algorithm proposed in [13] (we will refer it as IMM05), and the Hidden Markov Model based algorithm described in [9] (we will refer it as NK09). IMM05 performs matching based on three aspects: the distance, the direction and a point's previous matching result, and is known to have a low time complexity. NK09 is well-known for its high accuracy and the ability to deal with noise.

For IMM05, we use the suggested settings in [13]: $\mu_d = 10, \alpha = 0.17, n_d = 1.4, \mu_a = 10$, $n_\alpha = 4$. For NK09, we use the following assignment : $\sigma = 4.1$, and $\beta = 5$. Preprocessing and optimization suggested in [9] are made to NK09. To further speed up the algorithm, roads that are more than 50 meters away from the GPS point are eliminated from the candidate set.

5.2 Comparison of Matching Efficiency

As shown in Figure 4, EM^3* is 4∼10 times faster than the NK09, and is even comparable to IMM05 when sampling interval is small. In addition, the matching speed of EM^3 is a little faster than IMM05 . With the increase of the sampling interval, the speeds of EM^3 and EM^3* begin to decrease due to the fact that less computation can be avoided.

The following reasons may contribute to the high efficiency of EM^3*:

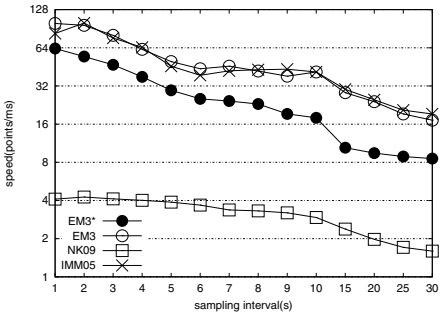


Fig. 4. Efficiency

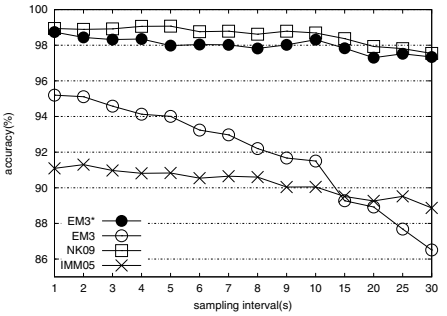


Fig. 5. Accuracy

- Computation of map-matching on a large part of GPS points is avoided. In most cases, the algorithm needs only to match the initial and the final points. Moreover, there is less shortest path calculation in EM^3* than NK09 which is quite time-consuming.
- The refined candidate management method is efficient. With the candidate generation method proposed in Section 4.2.1, the number of candidates is usually quite small.

5.3 Comparison of Matching Accuracy

As shown in Figure 5, the matching accuracy of EM^3* is quite comparable to NK09, and much higher than IMM05. Meanwhile, the accuracies of all these algorithms decrease with the increase of sampling interval as the deviation between the real route of the moving object and the trajectory represented by polyline becomes larger.

In particular, the matching accuracy of EM^3 decreases significantly with the increase of sampling interval. This is mainly because EM^3 simply choose the nearest vertex as the real passed one which is less accurate when the distances between the trajectory and passed vertices become larger. In contrast, EM^3* maintains all the possible paths, and eventually chooses the best one which makes it less sensitive to the increase of sampling interval. Several reasons may contribute to the high accuracy of EM^3*:

- The refined multi-hypothesis model is used to effectively maintain a set of candidate paths and eventually choose the best one.
- EM^3* is less dependent on the geometric detail of the edge, e.g., the proposed transition probability measurement method needs no projection to edges, which makes it robust to edge measurement errors.

6 Conclusion and Future Work

In this paper, we investigate the problem of how to efficiently perform map matching, and propose two novel algorithms to meet the requirement of high efficiency and accuracy posed by emergency scheduling and commanding . The proposed algorithms perform map-matching based on intersections that the moving object passes by, and are robust to edge measurement error and edge simplification. This feature also enables them to work with much smaller storage consumption. We conduct experimental study on real dataset to evaluate the performance of proposed algorithms which shows that EM^3* is both efficient and accurate.

In the future work, we plan to further improve the algorithm to better process GPS data of low sampling rate.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (Nos. 61202064, 91124001), the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA06020600), and the Knowledge Innovation Program of the Chinese Academy of Sciences (No. KGCX2-YW-174).

References

1. Liu, K., Deng, K., Ding, Z., Li, M., Zhou, X.: Moir/mt: Monitoring large-scale road network traffic in real-time. PVLDB 2, 1538–1541 (2009)

2. Li, M., Zhang, Y., Wang, W.: Analysis of congestion points based on probe car data. In: 12th International IEEE Conference on Intelligent Transportation Systems, ITSC 2009, pp. 1–5 (2009)
3. Gonzalez, H., Han, J., Li, X., Myslinska, M., Sondag, J.P.: Adaptive fastest path computation on a road network: a traffic mining approach. In: VLDB, pp. 794–805 (2007)
4. Chen, Y., Xiao, D.: Real-time traffic management under emergency evacuation based on dynamic traffic assignment. In: IEEE International Conference on Automation and Logistics, ICAL 2008, pp. 1376–1380 (2008)
5. Ding, Z., Deng, K.: Collecting and managing network-matched trajectories of moving objects in databases. In: Hameurlain, A., Liddle, S.W., Schewe, K.-D., Zhou, X. (eds.) DEXA 2011, Part I. LNCS, vol. 6860, pp. 270–279. Springer, Heidelberg (2011)
6. White, C.E., Bernstein, D., Kornhauser, A.L.: Some map matching algorithms for personal navigation assistants. Transportation Research Part C: Emerging Technologies 8(1-6), 91–108 (2000)
7. Greenfeld, J.S.: Matching gps observations to locations on a digital map. In: Transportation Research Board. Meeting, Washington, D.C (2002)
8. Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y.: Map-matching for low-sampling-rate gps trajectories. In: GIS, Seattle, Washington, pp. 352–361 (2009)
9. Newson, P., Krumm, J.: Hidden markov map matching through noise and sparseness. In: GIS, Seattle, WA, USA, pp. 336–343 (2009)
10. Zheng, K., Zheng, Y., Xie, X., Zhou, X.: Reducing uncertainty of low-sampling-rate trajectories. In: ICDE, Washington, DC, USA, pp. 1144–1155 (2012)
11. Pink, O., Hummel, B.: A statistical approach to map matching using road network geometry, topology and vehicular motion constraints. In: IEEE ITSC, pp. 862–867 (2008)
12. Wenk, C., Salas, R., Pfoser, D.: Addressing the need for map-matching speed: Localizing globalb curve-matching algorithms. In: SSDBM, Washington, DC, USA, pp. 379–388 (2006)
13. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: VLDB, Trondheim, Norway, pp. 853–864 (2005)
14. Quddus, M.A., Ochieng, W.Y., Noland, R.B.: Current map-matching algorithms for transport applications: State-of-the art and future research directions. Transportation Research Part C: Emerging Technologies 15(5), 312–328 (2007)
15. Syed, S., Cannon, M.E.: Fuzzy logic based-map matching algorithm for vehicle navigation system in urban canyons. In: National Technical Meeting of The Institute of Navigation, San Diego, CA, pp. 982–993 (2004)
16. Reid, D.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control 24(6), 843–854 (1979)
17. Pyo, J.-S., Shin, D.-H., Sung, T.-K.: Development of a map matching method using the multiple hypothesis technique. In: Proceedings of 2001 IEEE Intelligent Transportation Systems, Oakland, CA, USA, pp. 23–27 (2001)
18. Liu, K., Li, Y., He, F., Xu, J., Ding, Z.: Effective map-matching on the most simplified road network. In: GIS, Redondo Beach, CA, USA, pp. 609–612 (2012)
19. Zhou, J., Golledge, R.: A three-step general map matching method in the gis environment: travel/transportation study perspective. International Journal of Geographical Information System 8(3), 243–260 (2006)
20. ACM SIGSPATIAL Cup 2012: Training data sets (2012), `http://depts.washington.edu/giscup/home`

An Empirical Study of Information Diffusion in Micro-blogging Systems during Emergency Events

Kainan Cui[1,2], Xiaolong Zheng[2,3], Daniel Dajun Zeng[2],
Zhu Zhang[2], Chuan Luo[2], and Saike He[2]

[1] The School of Electronic and Information Engineering,
Xi'an Jiaotong University, Xi'an, China
[2] The State Key Laboratory of Management and Control for Complex Systems,
Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China
[3] Dongguan Research Institute of CASIA, Cloud Computing Center,
Chinese Academy of Sciences, Songshan Lake, Dongguan, 523808, China
kainan.cui@live.cn, xiaolong.zheng@ia.ac.cn

Abstract. Understanding the rapid information diffusion process in social media is critical for crisis management. Most of existing studies mainly focus on information diffusion patterns under the word-of-mouth spread mechanism. However, to date, the mass-media spread mechanism in social media is still not well studied. In this paper, we take the emergency event of Wenzhou train crash as a case and conduct an empirical analysis, utilizing geospatial correlation analysis and social network analysis, to explore the mass-meida spread mechanism in social media. By using the approach of agent-based modeling, we further make a quantativiely comparison with the information diffusion patterns under the word-of-mouth spread mechanism. Our exprimental results show that the mass-meida spread mechanism plays a more important role than that of the word-of-mouth in the information diffusion process during emergency events. The results of this paper can provide significant potential implications for crisis management.

Keywords: Information diffusion, opinion dynamic, emergency response, social media, micro-blogging systems.

1 Introduction

The rapid diffusion of information and opinions through social media is affecting the development of crisis situations [1-3]. Many researches show that social media has great potential for improving the situational awareness during emergency situation, such as mining the actionable information for earthquake [4], tracking the transmission rate and studying the response behavior for Influenza outbreak [5, 6]. However, despite those opportunities, the social media may also contribute to the effective broadcasting of rumors and extreme opinions, which will lead to huge social and economic damage [7].

Generally, there are two main sources, including the word-of-mouth and the mass-media[8], for us to obtain information. Word-of-mouth information spreading via

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 140–151, 2013.

online social network is considered as the main feature of the social media. Researches on the this mechanism have exploded in recent years [9]. The affecting factors of the word-of-mouth information spreading mechanism include social network structure[10, 11] , geographic distance [12], opinion leader [13] and so on. On the other hand, there are also evidences for the mass-media information spreading in social media. A recent study found that micro-blogging resembles a traditional news media more closely [14]. In another word, information diffusion in social media could also follow the mass-media information spreading mechanism. Understanding the information spreading mechanism has several implications for emergency management. From the perspective of countering rumors, it will not only provide us tools to predict the rumor dynamics, but also guide us on how to reduce the loss. To the best of our knowledge, few works have been done to empirically examine the information spreading mechanism in social media during emergency events.

To fill the research gaps, we conducted a case study to investigate how information was spread during the 2011 Wenzhou train crash through the Sina Weibo, a popular Chinese micro-blogging system. In particular, we performed cluster analysis on the diffusion outcomes at first. Then we applied an agent-based model to simulate the temporal trends of diffusion. Our results indicate that the mass-media spreding mechanism could better reproduce the temporal trends of information diffusion than the word-of-mouth spreding mechanism.

The rest of this paper is organized as follows. In Section 2, we begin with a brief introduction to the Wenzhou train crash dataset and intuitions obtained from the dataset. Motived by the intuitions from Section 2, we perform empirical analysis and report results in Section 3. Both evidences for word-of-mouth spreading mechanism and mass-media spreading mechanism were found in this section. In Section 4, we further applied an agent-based Bass model to simulate the temporal trends of our dataset. The simulation results suggest that the mass-media spreading mechanism played a more important role in social media during emergency events than word-of-mouth spreading mechanism. We conclude our paper by summarizing the findings and discussing several key issues in Section 5.

2 The Dataset

“Wenzhou train crash” accident happened at night of July 23, 2011 in Zhejiang Provience. This event caused 40 deaths and over 100 people injured. Micro-blogging systems played a crucial role in the information sharing and dissemination during the event. More than 5.3 million messages were posted to microblog about this event[15].

Poll system on the Sina Weibo gives us an excellent way to study the information spreading mechanisms during emergency events. The features of poll system make each poll in the platform a perfect diffuison process. At first, user can only paticipate specific poll through accessing certain URL. Second, each user is allowed to post once. There are two ways to spread a poll URL in the Micro-blogging platfrom. User sharing is the first one, which could be treated as word-of-mouth spreading mechanism. Users could also access certain poll from the new polls ranking and the hot polls ranking in the front page. This process is similar to the mass-media

spreading mechanism. Our dataset is a poll titled "Will you still support China High-speed Train?" The poll started at 15:05, July 26, 2011 and closed by 15:04, August 2, 2011. There are 2071 voters totally. By applying a customized crawler, we collected all vote records with corresponding user profiles and following relationships.

The original poll has five options which could be easily classified into positive opinions, negative opinions and neutral opinions. After the division, we have 645 users who support CHT (China High-speed Train), 773 users who are against the CHT, 653 users hold neutral views. The exsiting of competive opinions in dataset provide us opportunity to study the difference among the diffusion outcomes of varisous opinions.

2.1 Social Network of Voters

The word-of-mouth spreading mechanism relies on the social network structure. In Micro-blogging platform, the social network is constructed by the following relationships. After removing the edges linking to users who did not vote, we have 3888 edges. We found that the social network is not fully-connected, which means the poll URL could not reach all voters via person-to-person spreading. Therefore, the word-of-mouth spreading mechanism could not explain the diffusion outcomes alone. As shown in Fig. 1, we visualized the giant component of voters' social network by representing voters as nodes and relationships among them as links (The figure was plotted by NodeXL http://nodexl.codeplex.com/). This network is the biggest weakly connected sub-graph of the whole social network. It contains 863 nodes and 3505 edges. The red nodes represent voters who support CHT. The blue nodes represent voters who oppose CHT. The green nodes represent voters who have neutral views.

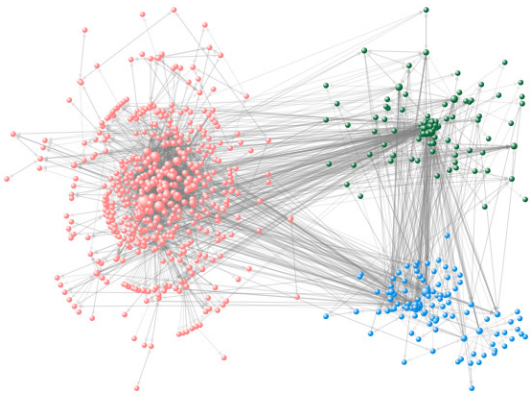


Fig. 1. The **structure of the giant component of voters' social network**, red nodes represent positive voters; blue nodes represent negative voters; green nodes represent neutral voters. The node size is in proportion to the out degree.

The exisiting of social network association is the essential condition for the word-of-mouth spreading mechanism. Therefore, if there is no social network association, then we can exclude the word-of-mouth spreading mechanism. We can observe an obvious cluster among positive voters. This intuition leads the social network association analysis in Section 3.1.

2.2 Geospatial Distribution of Voters

Geospatial distribution is another perspective of our dataset. The user profile contains the location information of most users. Under the assumption that people are more concerned about local events and have more willingness to post opinions on Micro-blogging, the areas near the accident spot will have more voters. After removing records outside mainland China or missing location information, we have 1882 records. Fig. 2 shows the geospatial variation of voters within province-level. This map intuitively presents that the voters are more concentrated in the accident spot and surrounding areas. We also found Beijing and Guangdong have more voters. Motivated by this result, we plan to quantify the geospatial correlation and identify the hot spot of voters in Section 3.2.

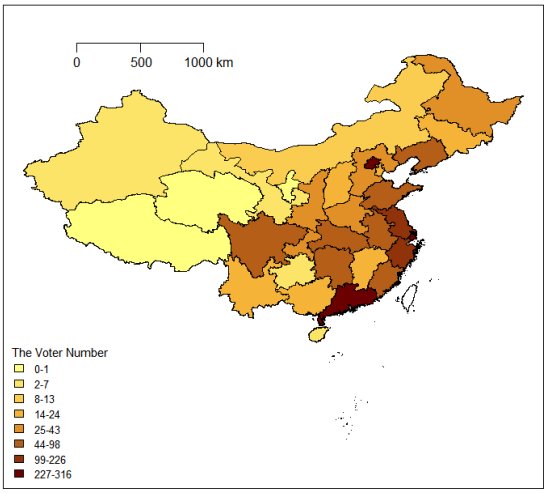


Fig. 2. Geospatial variation of diffusion outcomes in mainland China

3 Empirical Analysis

3.1 Social Network Association Analysis

Social Network Association of all Voters: We choose global cluster coefficient to analyze the whole voter network. Global cluster coefficient [16] is a measure to indicate the association degree of a graph and is defined as follows:

$$C = \frac{1}{n}\sum_{u \in G} \frac{T(u)}{deg(u)(deg(u)-1)} \quad (1)$$

Where G is the graph, n is the number of nodes in G, $T(u)$ is the number of triangles through node u, and $deg(u)$ is the number of neighbors of u.

As shown in Table 1, the global cluster coefficient of our voter network is a little bigger than the general network of Sina microblog. This result shows that the association level of our voter network is similar with the general network.

Table 1. A comparison of cluster degree between the general network of Sina microbog and the voter netwrok of our dataset

Network	APL	Diameter	Cluster Coefficient
General network[17]	4.0	12	0.21
Voter network	4.43	13	0.22

APL is the average path length; APL and Diameter is computed for the giant component of network

Social Network Association of Voters with Different Opinions: A commonly used method for social network association analysis among different categories is to construct a random mixing network and then compare the edges number in different categories[18]. In our dataset with 3 opinions, there are 9 types of edges (e.g., positive-positive edges, positive-negative, positive-neutral, etc.). Chi-square tests were used on the categorical data in order to assess whether there is a significant difference between the real network and the random mixing network.

Table 2 shows the comparison result between real network and random mixing network. From this table, we obtain the following observations: 1) the numbers of edges with same opinion are always greater than the numbers of edges with opposite opinion. For example, positive-positive edges (2591) are more than positive-negative edges (107), negative-negative edges (246) are more than negative-positive (97) edges, and so on; 2) the number of positive-positive edges from real network is much higher than the expected number; 3) the number of negative-negative edges and the number of neutral-neutral edges form real network are lower than the expected number. The results indicate strong social network association exists among positive voters, while the association level among negative voters and neutral voters are lower than random mixing network.

Table 2. A comparison of edges numbers between real network and random mixing network

follower \ followee	Positive	Negative	Neutral
Positive	**2591**	107	182
Expected	376.7224	452.1838	381.9871
Negative	97	246	181
Expected	452.1838	541.2185	457.7923
Neutral	142	157	185
Expected	381.9871	457.7923	386.1327

(Chi-squared = 1722.219, df = 4, p-value < 2.2e-16).

Expected is the number of edges in random mixing network.

3.2 Geospatial Association Analysis

Geospatial Association of all Voters: "Local Moran's I" [19] is selected to identify the geospatial hot spot for all voters. It is defined as follows:

$$I_i = \frac{n(x_i - \bar{x})\sum_{j=1}^{n} w_{ij}(x_j - \bar{x})}{\sum_{j=1}^{n}(x_j - \bar{x})^2} \tag{2}$$

Where x_i and x_j are the diffusion outcomes at area i and j, $\bar{x}$ is the average value of all areas in the entire region, w_{ij} is the weight of the spatial neighborhood relationship, and n is the population size.

Fig. 3 shows the cluster map of local Moran' I (The map was plotted by GeoDA software https://www.geoda.uiuc.edu/). A cluster is composed by locations which are more similar to its neighbors. The High-High locations refers to hot spot areas where the number of voters is higher than average, whereas the Low-Low locations refers to cool spot areas where the number of voters is lower than average. From Fig. 3, we can observe Shanghai, Zhejiang and Jiangsu as a hot spot, Xinjiang as a cool spot. These results indicate that the hot spot of voters is near the incident spot, while the cool spot of voters is far form the incident spot.

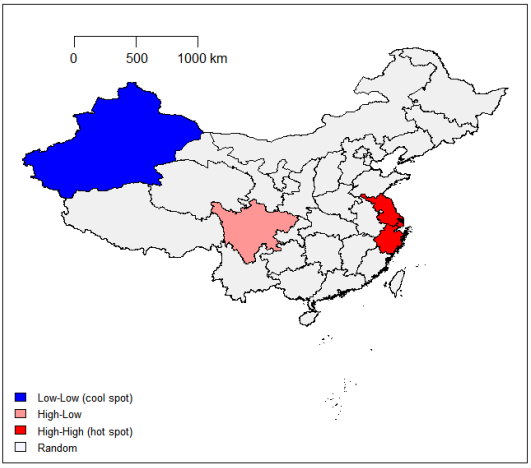


Fig. 3. Cluster map of diffusion outcomes based on Local Moran' I

Geospatial Association of Voters with Different Opinions: Global Moran's I [19] is selected to quantify the geospatial correlation for voters with different opinions. It is calculated as follows:

$$I = \frac{n\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(x_i - \bar{x})^2} \tag{3}$$

Where x_i, x_j, $\bar{x}$ and w_{ij} have the same meaning as in Equation (2). It is worthy to note that the global Moran's I is calculated by averaging the local Moran's I in each area.

The statistics are summarized in Table 3. From this table, we obtain the following observations 1) the geospatial distribution of all voters has positive geospatial association; 2) the geospatial distribution of positive voters is nearly random; 3) the geospatial distribution of negative voters and the geospatial distribution of neural voters have positive geospatial association.

Table 3. A comparison for the geospatial association of voters with different opinions

Opinions	Moran's I	999 MC simulation			
		E(i)	S_d	Z-score	P-value
Positive	0.092283	-0.0323	-0.0352	1.4447	0.093
Negative	0.271442	-0.0323	0.1075	2.7739	0.012
Neutral	0.228925	-0.323	0.1027	2.5496	0.019
All	0.20936	-0.0357	0.1008	2.4301	0.024

3.3 Summary of Empirical Analysis

In this Section, we performed empirical analysis to quantify the intuitions obtained in Section 2. In particular, social network association analysis and geospatial association analysis were conducted respectively. The results of association analysis are summarized in Table 4.

For all voters, we found a medium cluster degree in social network and a positive correlation in geospatial distribution; for positive voters, we found a high cluster degree in social network and a nearly random correlation in geospatial distribution; for negative and neutral voters, we found a low cluster degree and a positive correlation in geospatial distribution. Although the geospatial distribution of the voter population expresses a positive correlation as a whole, which could be better explained by the mass-media information spreading mechanism, we cannot rule out the word-of-mouth information spreading mechanism based on the results thus far. The high cluster degree of positive voters provides evidence for the word-of-mouth information spreading mechanism.

Table 4. A summary of results about social network association analysis and geospatial association analysis

Opinions	social network cluster degree	geospatial correlation
Positive voters	high	nearly random (0.09)
Negative voters	low	positive (0.27)
Neutral voters	low	positive (0.23)
All voters	medium	positive (0.21)

4 Experiments

In this section, we plan to further analyze the temporal dynamics of the diffusion process through simulation. We will first present the time series of the diffusion dynamics and discuss the design of experiments. In Section 4.1, we will describe an agent based model inspired by the Bass model. In Section 4.2, we will show the experiments results.

Because the voter's social network is not fully-connected, we simulate the information diffusion in the giant component of the voters' social network. Fig. 4 shows the time series for daily new voter numbers in the giant component of voters' social network. The plot shows a sharply increasing during the second day and a rapidly declining during the following period.

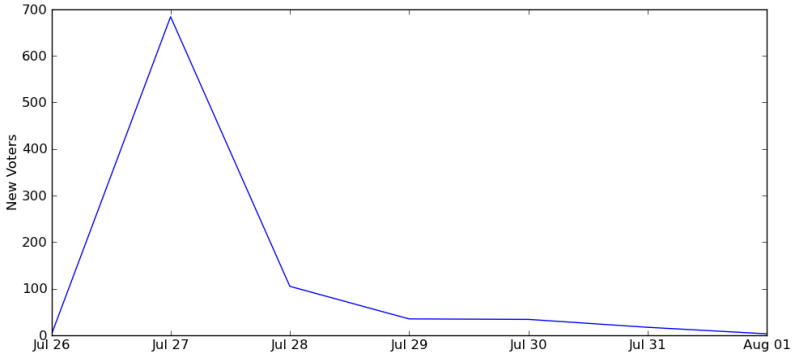


Fig. 4. Time series plots of new voters in the giant component of voter's social network

4.1 Models

The Bass model [8], which was originally developed to model the diffusion of new products in marketing, is defined as follows:

$$\frac{f(t)}{1-F(t)} = p + qF(t) \tag{4}$$

Where $f(t)$ is the rate of change of the adopted fraction, $F(t)$ is the adopted fraction, p is the coefficient for mass media effect, q is the coefficient for word-of-mouth effect.

In order to leverage the information of real-world network structure, we implement the Bass model in the framework of agent based modeling [20]. In this model, each agent is affected only by its neighbors, instead of the entire population. We introduce the model as follows: 1) the states of agents are classified into unaware state and aware state; 2) mass-media spreading mechanism: at each time step, every unaware agent has a possibility α to turn into aware state; 3) word-of-mouth spreading mechanism: at each time period, every aware agent attempts to affect their neighbors

with transmission rate β; 4) At time step t=0, only 1 agent is aware; 5) the network structure is initialized according to the real-world network.

There are totally 8 experiments scenarios. Half of them are designed for the word-of-mouth spreading mechanism and half of them are designed for the mass-media spreading mechanism. In the scenarios for word-of-mouth spreading mechanism, the parameter α was set to 0, and the parameter β was set to 0.6, 0.7, 0.8 and 0.9 respectively. In the scenarios for mass-media spreading mechanism, parameter β was set to 0, and the parameter α was set to 0.6, 0.7, 0.8 and 0.9 respectively. We will run 50 times for each scenario. The results will be averaged and displayed in Section 4.2.

4.2 Results

We will first present the simulation results for the word-of-mouth spreading mechanism. Then will present the simulation results for the mass-media spreading mechanism. As Fig. 5 shown, word-of-mouth spreading mechanism could not reproduce the temporal trends of the diffusion process alone.

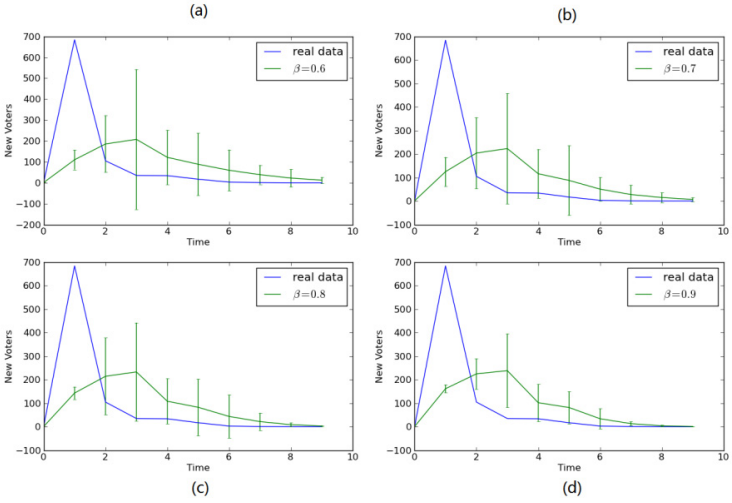


Fig. 5. Simulation result of word-of-mouth spreading mechanism. Four scenarios with different value of beta are displayed, each run 50 times and averaged. (a) beta=0.6. (b) beta=0.7. (c) beta=0.8. (d) beta=0.9.

As shown in Fig. 6, mass-media spreading mechanism could well reproduce the temporal trends of the diffusion process. These results indicate that the mass-media spreading mechanism play a more important role in our dataset.

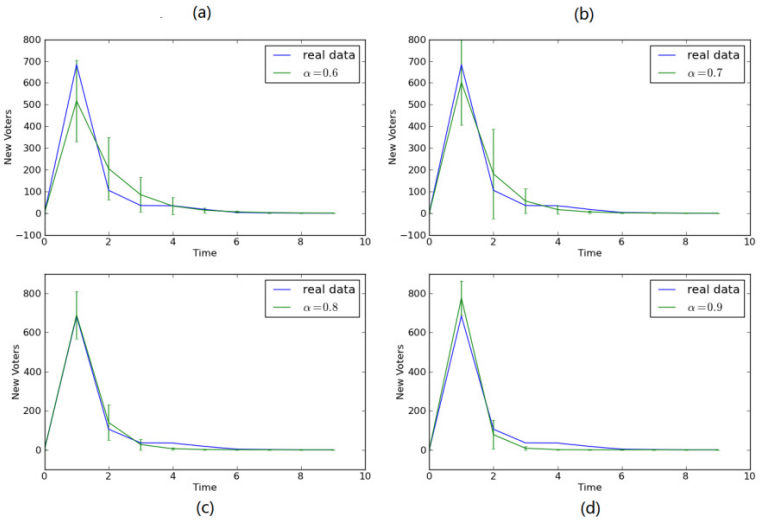


Fig. 6. Simulation result of mass-media spreading mechanism. Four scenarios with different value of alpha are displayed, each run 50 times and averaged. (a) alpha=0.6. (b) alpha =0.7. (c) alpha =0.8. (d) alpha =0.9.

To quantify the results in Fig.5 and Fig.6, the person correlation coefficient was calculated between the simulation output and the original data. As summarized in Table 5, the mass-media spreading mechanism has is obviously outperform word-of-mouth spreading mechanism.

Table 5. A comparison of correlation between the word-of-mouth spreading mechanism and the mass-media spreading mechanism

Mechanism	α	β	Person correlation	P-value
Mass-media	0.6	0	0.96761	$P<0.001$
	0.7	0	0.98735	$P<0.001$
	0.8	0	0.99726	$P<0.001$
	0.9	0	**0.99791**	$P<0.001$
Word-of-mouth	0	0.6	0.16003	0.510
	0	0.7	0.21718	0.372
	0	0.8	0.28764	0.238
	0	0.9	0.34966	0.159

5 Conclusion

In our research, we have investigated a real-world information diffusion case in Micro-blogging during the Wenzhou train crash event. Our study reveals that mass-media spreading mechanism can better explain the empirical data than the word-of-mouth spreading mechanism, which is considered as the main feature of information diffusion in social media. One possible explanation could be that the credibility of the

information sources is more important during emergency events. According to the Bandwagon effect, people are likely to believe the choice of public. The hot topic ranking system as a mechanism to reflect the public choice will play the role like the mass media. If this assumption is true, we should focus on how to design ranking algorithms and borrow experience form mass media to improve the information credibility.

Another interesting phenomena are further found in this paper. We obtain that a high cluster degree in social network is related to a nearly random correlation in geospatial distribution and a low cluster degree in social network is related to a positive correlation in geospatial distribution. This phenomenon could be explained by the online communication, which break the physical boundary for information sharing and extend the association from offline world to the online world.

One of the limitations in this study is that we did not consider the two spreading mechanisms at the same time. However, this work still shed light on the information diffusion through Micro-blogging during emergency events. Further empirical work is needed to verify our findings.

Acknowledgments. The authors would like to thank Zhaonan Zhu and Jiaqi Liu for their helpful suggestions. This work was supported in part by the following grants: The National Natural Science Foundation of China, Grant No. 91124001, 71103180, 91024030 and by the Ministry of Health under Grant No. 2012ZX10004801.

References

1. Adam, N.R., Shafiq, B., Staffin, R.: Spatial Computing and Social Media in the Context of Disaster Management. IEEE Intelligent Systems 27, 90–96 (2012)
2. Daniel, Z., Hsinchun, C., Lusch, R., Shu-Hsing, L.: Social Media Analytics and Intelligence. IEEE Intelligent Systems 25, 13–16 (2010)
3. Zeng, D., Wang, F.-Y., Carley, K.M.: Guest Editors' Introduction: Social Computing. IEEE Intelligent Systems 22, 20–22 (2007)
4. Tyshchuk, Y., Wallace, W.A.: Actionable Information during Extreme Events – Case Study: Warnings and 2011 Tohoku Earthquake. In: Conference Actionable Information during Extreme Events – Case Study: Warnings and 2011 Tohoku Earthquake, pp. 338–347 (2012)
5. Culotta, A.: Towards detecting influenza epidemics by analyzing Twitter messages. In: Proceedings of the First Workshop on Social Media Analytics, pp. 115–122. ACM, Washington D.C (2010)
6. Cui, K., Cao, Z., Zheng, X., Zeng, D., Zeng, K., Zheng, M.: A Geospatial Analysis on the Potential Value of News Comments in Infectious Disease Surveillance. In: Chau, M., Wang, G.A., Zheng, X., Chen, H., Zeng, D., Mao, W. (eds.) PAISI 2011. LNCS, vol. 6749, pp. 85–93. Springer, Heidelberg (2011)
7. Il-Chul, M., Oh, A.H., Carley, K.M.: Analyzing social media in escalating crisis situations. In: Conference Analyzing Social Media in Escalating Crisis Situations, pp. 71–76 (2011)
8. Bass, F.M.: Comments on "A New Product Growth for Model Consumer Durables The Bass Model". Management Science 50, 1833–1840 (2004)

9. Zheng, X., Zhong, Y., Zeng, D., Wang, F.-Y.: Social influence and spread dynamics in social networks. Front. Comput. Sci. 6, 611–620 (2012)
10. Kim, M., Xie, L., Christen, P.: Event Diffusion Patterns in Social Media. In: Conference Event Diffusion Patterns in Social Media (2012)
11. Zheng, X., Zeng, D., Li, H., Wang, F.: Analyzing open-source software systems as complex networks. Physica A: Statistical Mechanics and its Applications 387, 6190–6200 (2008)
12. Toole, J.L., Cha, M., González, M.C.: Modeling the Adoption of Innovations in the Presence of Geographic and Media Influences. PLoS ONE 7,e29528 (2012)
13. Watts, D.J., Dodds, P.S.: Influentials, Networks, and Public Opinion Formation. Journal of Consumer Research 34, 441–458 (2007)
14. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: Proceedings of the 19th International Conference on World Wide Web, pp. 591–600. ACM, Raleigh (2010)
15. Xu, X.: Internet Facilitated Civic Engagement in China's Context: A Case Study of the Internet Event of Wenzhou High-speed Train Accident (2011)
16. Watts, D.J., Strogatz, S.H.: Collective dynamics of /'small-world/' networks. Nature 393, 440–442 (1998)
17. Pengyi, F., Pei, L., Zhihong, J., Wei, L., Hui, W.: Measurement and analysis of topology and information propagation on Sina-Microblog. In: Conference Measurement and Analysis of Topology and Information Propagation on Sina-Microblog, pp. 396–401 (2011)
18. Thelwall, M.: Homophily in MySpace. Journal of the American Society for Information Science and Technology 60, 219–231 (2009)
19. Moran, P.A.: The interpretation of statistical maps. Journal of the Royal Statistical Society. Series B (Methodological) 10, 243–251 (1948)
20. Li, X., Mao, W., Zeng, D., Wang, F.-Y.: Agent-Based Social Simulation and Modeling in Social Computing. In: Yang, C.C., et al. (eds.) ISI Workshops 2008. LNCS, vol. 5075, pp. 401–412. Springer, Heidelberg (2008)

Cluster-By: An Efficient Clustering Operator in Emergency Management Database Systems

Peng Sun[1], Yan Huang[2], and Chengyang Zhang[3]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, China
sunpeng@iscas.ac.cn
[2] University of North Texas, Denton, TX, U.S.A
huangyan@unt.edu
[3] Teradata Inc., El Segundo, CA, U.S.A
chengyang.zhang@teradata.com

Abstract. Database management systems (DBMS) have been widely used to efficiently store, manage and analysis large emergency management data. Despite the popularity of clustering as a general data mining method, current emergency management database systems lacked a unified and convenient way to support in-database clustering. In this paper we promote the advantages of integrating clustering into databases and propose a new *Cluster-by* SQL extension. We formally define the syntax and semantics of the *Cluster-by* clause, illustrate its query plan node in database engine and present two data preprocessing rules. Then we explore the query optimization opportunities, present a novel framework for multiquery optimization and define the cost model for multi-query scheduling. We also introduce DBSCAN-based Shrink and Expand algorithms to utilize the historical clustering results and present a heuristic cost model. To demonstrate the integration of the extension with existing DBMSs, we implemented the *Cluster-by* extension in PostgreSQL. We performed experiments on real data sets in PostgreSQL. Results show that *Cluster-by* extension is useful, the multiquery optimization techniques proposed are efficient.

Keywords: Cluster-by operator, SQL, in-database clustering.

1 Introduction

Database management systems (DBMS) have been widely used to efficiently store , manage and analysis large emergency management data, including spatial and nonspatial dataset [6, 12]. In-database mining for automatic discovery of interesting and potentially useful patterns becomes increasingly attractive because it avoids moving large datasets in-and-out-of a DBMS. As one of the most popular data mining methods, clustering [11] can be employed to find new geographic regions from a database of point of interests (POIs) [5].

However, current DBMSs lack a unified and convenient way to support **in-database clustering**. Although many existing database system has its own proprietary implementation of clustering, none of them has provided clear syntax and

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 152–164, 2013.

semantics for clustering. Instead, application developers have to rely on non-native software packages to perform the task of clustering.

In this paper, we propose a new **Cluster-by** SQL extension for DBMS to allow in-database clustering. Integrating the clustering semantics into DBMS provides many benefits. First, instead of moving large datasets stored in the database to an external clustering module, the clustering algorithm is performed directly in the database to eliminate network overhead. Second, the integrated approach allows query optimization opportunities that are otherwise not possible for stand alone clustering module. Third, clustering inside a DBMS can take advantage of many built-in features such as concurrency control, durability, and secure access mechanisms [10].

In our previous work [16], we recognized the advantages of *Cluster-by* extension and demonstrated the usage of such extension from a practical point of view. However, we did not provide clear definitions of the syntax and semantics. We did not address the issues of query optimization and implementation either. This paper is a logical step forward and brings the following novel contributions.

- We propose a new **Cluster-by** SQL clause for spatial and non-spatial DBMSs and provide clear definitions of syntax and semantics.
- We explore the **query optimization** opportunities and present a novel framework for multi-query optimization. We also propose an efficient multi-query scheduling algorithm.
- We design the **Shrink, Expand** algorithm for a representative density-based algorithm that utilizes the historical clustering results.
- We implement the *Cluster-by* extension in PostgreSQL. We performed several experiments on real data sets in PostgreSQL. Results show that multi-query optimization techniques proposed are efficient

The remainder of the paper is organized as follows. We review the related work in Section 2. Then we introduce Cluster-by extension and define the semantics of the extension in Section 3. In Section 4, we explore query optimization opportunities and propose a novel multi-query optimization framework. We describe Shrink and Expand algorithms in Section 5. We then discuss the implementation details of the extension in PostgreSQL in Section 6, and present the experimental results in Section 7. Finally we conclude the paper in Section 8.

2 Related Work

2.1 In-Database Clustering

Although clustering problem has been studied extensively, few work was dedicated to the integration of clustering algorithms in SQL and its extensions in DBMSs. Ordonez [10] presented three SQL implementations of k-means clustering in an object relational DBMS: standard, optimized, and incremental. Given parameters of table Y with d selected columns of numerical values and the desired number of clusters, Ordonez used a SQL generator to automatically generate a set of SQL statements to

implement the k-means clustering algorithm. However, this work only explored the approach of implementing clustering outside the database using existing SQL language. By contrast, our *Cluster-by* operator is directly integrated into the DBMS.

Some commercial SDBMSs included stored procedures or functions for clustering. In Oracle Spatial 11j's spatial analysis and mining package [1], a cluster computing function "SDO SAM.SPATIAL CLUSTERS()" is provided. It returns a set of "SDO REGION" objects that represent the bounding rectangles of each cluster. However, the semantics of the function is very limited and the query optimization opportunity of a tight integration is not explored.

From the perspective of grouping objects with similar, imprecise or approximate values, the purpose of our *Cluster-by* clause is close to Similarity-based *Group-by* (SGB) operator proposed by Silva et al.[13]. SGB operator uses group size and compactness to compute the groups instead of applying clustering algorithms. It heavily relies on the two parameters: s denoting the maximum distance between any two neighboring elements, and d denoting the maximum distance between any two elements in a group. The use of SGB operator is restricted when these parameters are not available beforehand. Li et. al. proposed generalized *Group-by* to enable fuzzy grouping using clustering method and integration with ranking [8]. The cost reduction technique is tired with the ranking and is not generalized. In our previous work [16], we have proposed an initial SQL extension to execute spatial clustering in spatial DBMS and demonstrated several motivating examples without implementation and performance testing.

2.2 Query Optimization

Query optimization is an important issue in traditional DBMS, and is even more crucial for SQL queries with *Cluster-by* extension because clustering algorithm is time- and space consuming. For queries with both join and "group by" operators, WP. Yan et al. [14, 15] proposed to interchange the order of grouping and join to reduce the cost of query processing and generate the optimized execution plan. The *group by push down* strategy reduces the number of rows participating in a join, and *group by pull up* strategy reduces the number of input rows to the "group by" clause. The interchange of the "group by" and join operators can only be performed when the following two conditions satisfy: (1) the joins are all foreign key joins, and (2) R_d (the table containing aggregation columns) grouping columns contain all the columns that join Ru (the table not containing aggregation columns). Silva et al. [13] further extended the *group by push down* and *group by pull up* strategies for similarity based Group-by (SGB) operator. However, the application of the above conditions on our *Cluster-by* clause is not appropriate because the non-algebraic nature of the cluster combination. As a result, extending the existing query optimization methods is non-trivial.

Another type of optimization method exploits the commonality of multiple queries and avoids the re-computing process. Multi-query optimization is frequently used in data warehouses where materialized views are created to optimize the execution of several simultaneous queries [7]. Our multi-query optimization framework uses similar ideas but with very different types of materialization and replacement strategy.

2.3 Variable Parameters DBSCAN

DBSCAN [3] is a density-based clustering algorithm. It finds regions with high density (defined by two parameters minimum number of points minPts within radius Eps) above a user give threshold as clusters. Li et. al. proposed a variable parameters DBSCAN algorithm called IPC-DBSCAN [4]. When minPts descreases, four possibilities happen to existing clusters: 1) No change, 2) Creation: new clusters emerge from noise, 3) Expanding: existing clusters expand to include some former border points, 4) Merge: some closely located clusters could be merged. When minPts increases, there are also four cases: 1) No change, 2) Removal, 3) Shrink, 4) Split. Different from this work, we discuss eight cases of changing (ε, minPts) combinations and propose three processing strategies. We also define merge conditions to guide the merge of two close clusters. Further more, our strategies are used by the multi-query optimization module. Martin Ester also proposed Shrink and Expand concepts in their incremental DBSCAN paper, addressing insertion or deletion of an object [2] from the dataset.

3 Cluster-by: Syntax and Semantics

3.1 Formal Syntax

Following the discussion in our work [16], we now present the formal syntax of the *Cluster-by* clause.

<table expression> ::= <*from clause*>[<*where clause*>]
[<*group by clause*>|**<Cluster by clause>**]
[<*having clause*>][<*window clause*>]

<Cluster by clause> ::= CLUSTER BY <clustering column reference list>
[USING <clustering algorithm function>]

<clustering column reference list> ::= <weighted column reference>
[{ <*comma*> <weighted column reference>}...]

<weighted column reference> ::= < *columnreference* >
|< *columnreference* >< *asterisk* >< *factor* >
|< *factor* >< *asterisk* >< *columnreference* >

<clustering algorithm function> ::= <clustering algorithm name>
<*left parent*> [<value list>] <*right parent*>

<clustering algorithm name> ::= <qualified clustering algorithm name>
| <user-defined clustering algorithm name>

<qualified clustering algorithm name> ::= KMEANS | DBSCAN | ...

<user-defined clustering algorithm name> ::= <identifier>

<value list> ::= <*value expression*> [{ <*comma*> <*value expression*>}...]

In the above specification, the term in italic, e.g. <*value expression*> or <*group by clause*>, has the same meaning as that defined in ISO/IEC 9075-2:2008.

The *Cluster-by* clause starts with ***CLUSTER BY*** keywords, which are followed by a list of weighted columns and the clustering algorithm applying on all these columns. The clustering algorithm can be any of the system defined ones or a user specified new algorithm. Each clustering algorithm has its own list of parameters and values. The whole distance of these weighted columns is defined as the weighted sum of each distance of participated column. Please note that in this definition, <*select list*> requires all selected columns to be specified within a <*set function specification*> (aggregation functions) which is different from <group by clause>. Common aggregation functions include *AVG, MAX,* and *COUNT* etc. The <*set function specification*> also includes spatial aggregate functions such as convexhull of a point set. The <*having clause*> has the same syntax for *Cluster-by* as that for *Group-by* in SQL 2008.

3.2 The Semantics of *Cluster-by* Clause

Definition 1*: Clustering by Multiple Columns
Given an input relation: A = {col_1, col_2,..., col_l} (for brevity we also use A to denote its tuple set), without loss of generality, we assume the first m columns, are chosen to be clustered. Let *CV* denote this column vector of size m, WCV denote the corresponding weight coefficient vector, and *DFV* denote distance function vector, and *CAF* denote the clustering algorithm function with parameters {p_1, p_2, ..., p_n}:

$$CV = \{col_1, col_2, ..., col_m\}$$
$$WCV = \{w_1, w_2, ..., w_m\}$$
$$DFV = \{ df_1, df_2, ..., df_m \}$$

We have the Cluster-By subclause:

CLUSTER BY $w_1 * col_1, w_2 * col_2, ..., w_m * col_m$
USING $CAF(p_1, p_2, ..., p_n)$

Then the overall distance function ODF can be defined as:

$$ODF = \Sigma\ (w_i * df_i\ (col_i)) \mid i = 1,...,m$$

And g groups/clusters obtained after applying clustering algorithm function CAF and ODF is defined as:

$$\xi_{A,CAF,ODF} = \{\ M_i \mid M_i \subseteq A,\ M_i \text{ is generated by CAF with ODF, } i=1,...,g\ \}$$

3.3 Query Plan Node for *Cluster-by* Clause

In this section, the plan node of *Cluster-by* Clause in the whole plan tree in database engine will be presented. Fig. 1a shows the abstract view of the *Cluster-by* plan tree.

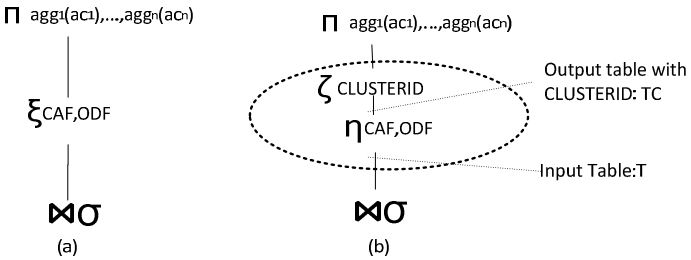


Fig. 1. Query Plan Node for *Cluster-by* Clause

We divide the *Cluster-by* operation (denoted by ξ) into two steps. First a *CLUSTERID* value is assigned to each input tuple using clustering algorithm (denoted by η). Then Group-by operator is performed on *CLUSTERID* to group each cluster (denoted by ζ), illustrated in Fig. 1b. The input table of *Cluster-by* is abstracted as T, and the output table of $\eta_{CAF,ODF}$ is abstracted as TC with a new attribute of *CLUSTERID*. We use $CA = \{cc_1, cc_2,...,cc_m\}$ to denote the clustered column set, $AA = \{ ac_1, ac_2,...,ac_n\}$ to denote the aggregated columns set, $TA = CA \cup AA$ to denote the attributes set of T, and $TCA = TA \cup \{CLUSTERID\}$ to denote the attributes set of *TC*. Then the above plan tree can be defined as a triple:

$$TC = \{T,C,P\}, \quad C = \xi_{CAF,ODF}, \quad P = \pi_{agg_1(ac_1),...,agg_n(ac_n)}$$

4 Multi-query Opitmization Framework

In this section, we will discuss query optimization methods for the proposed *Cluster-by* extension. For single query, *push down* and *pull up* strategies have been proposed by WP. Yan et al. [14, 15] to optimize queries with both *Group-by* and join operators. Traditional multi-query optimization techniques try to utilize the common subexpressions among queries to find a global optimal plan. In our approach, we aim at reducing the cost of clustering process by utilizing historical results and statistics.

4.1 General Framework

A typical single *Cluster-by* SQL execution without optimization framework is illustrated in Fig. 2a. In this figure, there are four steps when executing a *Cluster- by* plan tree. Firstly, the input tuple set T is prepared as input to Clustering Algorithm Execute Engine(*CAEE*). *CAEE* then choose a appropriate algorithm like DBSCAN to execute. The output of *CAEE* is a tuple set *TC*, which is basically T argumented with *CLUSTERID* attribute. The tuple set *TC* is transferred to the module *Group-by* on *CLUSTERID*(*GC*) and finally to the Aggregation Execution Module (*AEM*). The last two steps are usually integrated as one which is similar to their counterpart in traditional DBMS.

Once the necessary conditions (such as maximum memory and tuples count that could be processed in memory) are satisfied, the clustering algorithm execute engine could employ the optimization framework to utilize the historical clustering results to reduce the cluster plan node executing time.

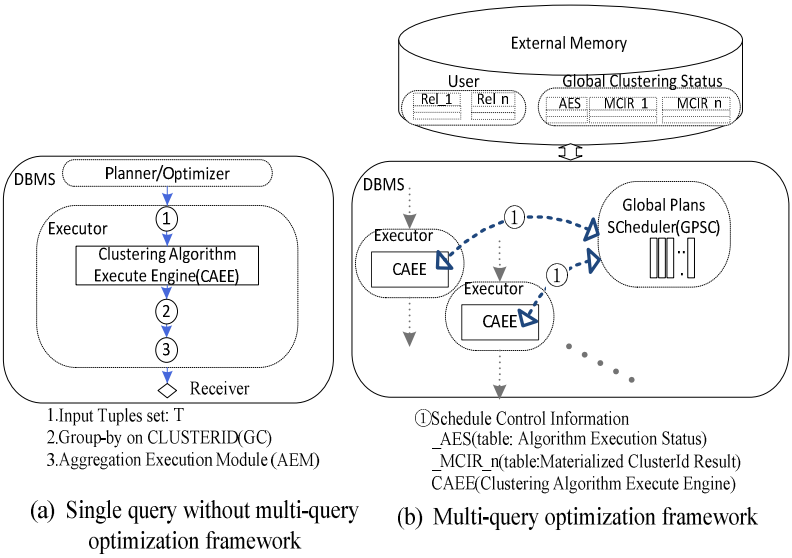


Fig. 2. Multi-query Framework Overview

Fig. 2b gives the multi-query optimization framework. The basic idea of this framework is to provide facilities to store historical (materialized) clustering results and statistics for future use. Those queries with same algorithms (yet different parameters) on the same input dataset can share the stored information. The framework mainly consists of two components: Global Clustering Status (*GCS*) and Global Plan Scheduler (*GPSC*).

Global Clustering Status(*GCS*) module stores the historical clustering results and statistics. Two sets of tables are used in this module: *_MCIR_x* and *_AES*. *MCIR* stands for Materialized ClusterId Result, *AES* stands for Algorithm Execution Status. There could exist many *_MCIR_*x tables, yet only one *_AES* table.

Global Plan Scheduler Global Plans Scheduler (*GPSC*) is responsible for scheduling execution requests of plan trees.

4.2 Cost Model

GPSC should know which history clustering result should be chosen to perform clustering or just uses the original algorithm (denoted as Brute Forth Algorithm). So we need two functions to estimate the cost of the brute forth algorithm and efficiency of the dependent algorithm, denoted as *BFF* and *EF* respectively. They are defined as:

$BFF = BFF$ (*_MCIR_x*, p_1, p_2,...,p_n)
$EF = EF$ (*_AES.AES_x*, *AES*)

BFF is used to estimate the cost without using any historical result, parameter p_1, p_2,...,p_n is as same as in function *CAF* in Section 3. *EF* function is used to estimate the cost when using a dependent *_AES.AES_x* for current *AES*.

5 Dependent DBSCAN

With the proposed framework, the next question is how to modify clustering algorithm and how to define *BFF* and *EF* function to utilize the framework. Now we use DBSCAN as an example to illustrate this process.

5.1 Algorithm

DBSCAN has two parameters, namely radius ε and minimum number of points minPts. Since each parameter can change in three directions: no changing, decrease and increase, there are eight (ε, minPts) combination of changement. Firstly we introduce a lemma:

Lemma 1: The increase of minPts or decrease of ε makes conditions of core point more strictly. And the decrease of minPts or increase of ε makes conditions of core object loosen.

Using **Lemma 1**, we can summarize the nine combinations into three cases when (ε´, minPts´) is changed to (ε, minPts):

1. (ε = ε´, minPts ≥ minPts´), (ε < ε´, minPts = minPts´) or (ε < ε´, minPts > minPts´)

When ε or minPts becomes more strict while the other has no changing or also becomes more stricly, noise is still noise. A formal cluster can: shrink, split or disappear. Each cluster has to be recalculated using its former clusters' members. We will use the *Shrink* algorithm on each cluster to form new cluster.

2. (ε < ε´, minPts < minPts´) or (ε > ε´, minPts > minPts´)

In this situation, one parameter becomes more strict while the other does not. So noises can be: noises, absorbed by a close cluster, or forms a new cluster. Members in a former cluster can be: noise or still in current cluster. Two clusters could be merged if they are close enough. We use original DBSCAN algorithm for the *Shuffle* algorithm in this paper.

3. (ε > ε´, minPts = minPts´), (ε = ε´, minPts < minPts´) or (ε > ε´, minPts < minPts´)

In this case, ε or minPts becomes more loosen while the other has no change or also becomes more loosen. In this situation, noise could be in a adjacent cluster, each cluster has to be recalculated with noise data to form possible expanded clusters. All noises should be checked to see whether new clusters could be formed by themselves. Then each pair of newly formed cluster should be tested to see whether to be merged or not. We will use *Expand* algorithm to be described shortly to handle this situation.

The basic idea of *Shrink* algorithm is to use DBSCAN algorithm on former clusters' members to discard noise. The design of *Expand* algorithm is to classify each noise data to its nearest cluster's absorber if possible. The absorber is defined as a cluster's core point within ε distance from this noise. If we got such absorber, expanding from this absorber only involves its cluster's tuples and noise data. If such expanding process could absorb other clusters' tuples, *expandToNoise* (as same as in DBSCAN) just stop this expanding process and let the following *Merge* algorithm do merge. If noises themselves could form new cluster, the function *expandToNoise* is also used to do such work. In theory, two clusters could be merged only if the minimal distance of each pair of these two clusters' core points is less than or equal to ε. Since if two core objects' distance is great than ε, we do not need compare with them. We introduce the *Common Area* to reduce core points involved for computing. The ***Common Area*** is defined as the intersect of two clusters' convex hull buffering with ε. The merge condition of two clusters is describes as followings:

At least there is one ClusterA's core point in *Common Area*, and at least there is one ClusterB's core point in *Common Area*, and at least there is one pair of two sets' core point whose distance is less than or equal to ε.

5.2 Cost Model Analysis

The BFF and EF function used in cost model for dependent DBSCAN is defined as:

$$BFF_{DBSCAN} = n \times (Pages_{idx\_i} \times rpc + Pages_{MCIR\_i} \times rpc + filteredTuples(\varepsilon) \times ctdc)$$

$$EF_{Shrink} = (1\text{-}noise/n) \times (Pages_{idx\_cid} + Pages_{cid}) \times rpc + numClusters \times avgClusterCount \times ctc + (1\text{-}noise/n) \times BFF_{DBSCAN}(\_MCIR\_x, MinPts, \varepsilon)$$

$$EF_{Expand} = noise/n \times BFF_{DBSCAN}(MCIR\ x, MinPts, \varepsilon) + Cost_{storeMiddleResult} + Cost_{computeIntersectArray} + Cost_{computeMergeArray}$$

The meanings of these symbols lists in Table 1.

Table 1. symbols abbreviation and its meanings

Abbreviation	Meanings
rpc	random_page_cost
ctc	cpu_tuple_cost
ctdc	cpu_tuple_distance_cost

6 Implementing Clustering in PostgreSQL

6.1 Implementing *Cluster-by* for Single Query

Parser Fig. 3a shows the modified cluster clause in gram.y. We also define a *clusterClause* in *SelectStmt*. It points to *ClusterVar* which has a column expression list and a functional call. Please note that in the gram.y definition, we use *CLUSTERING* as the keyword instead of *Cluster* because the later has already been reserved for other purposes in the system.

Query. The support of Cluster-by clause in the query phase is achieved through modified *Query* and *TargetEntry* structure in PostgreSQL. We also define a new data structure called *ClusterClause*. It is illustrated in Fig. 3b.

Optimizer. We define a new plan node *Cluster* to represent the first step (η) shown in Fig. 1b. The second step (ζ) shown in Fig. 1b is implemented using the *Agg* plan node, actually it depends on the plan tree originally generated by postgreSQL. Fig. 3c shows how to combine these two nodes to represent the extension in the optimizer.

Executor. Fig. 3d shows the new defined data structures in the executor phase. *ClusteringState* is the main state node in PlanState tree. Just like *Sort* node, *Cluster* node also need fetch all the tuples from its child node and then do clustering algorithm on all these tuples, once it's finished, *cluster_Done* flag is set *TRUE*, and the parent node could fetch each result tuples from *Cluster* node.

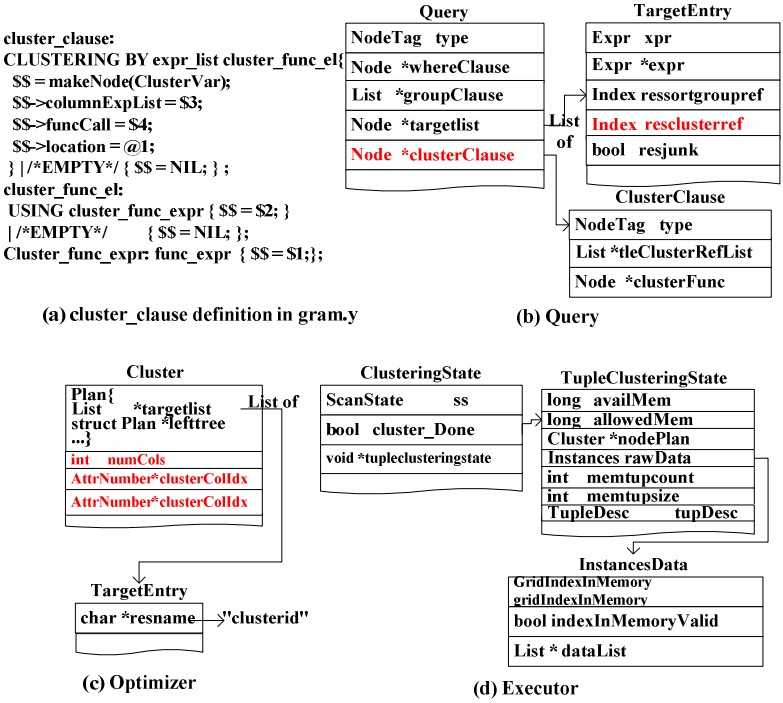


Fig. 3. Modified Data Structure in PostgreSQL

6.2 Implementing Multi-query Optimization Framework

The implementation of multi-query optimization framework mainly consists of two components: GPSC and backbend's ExecCluster main function. Two data structures are defined in GPSC: *GpscShmemStruct*, *GPSCSchedulerInfo*. When postmaster starts, it initialize GpscShmemStruct, fork the process GPSC, let global variable gpscScheduler point to GpscShmemStruct, that is in shared memory. The field gpsc pid is the process id used by GPSC process itself. BackendRequest is used to store each backend's request information, it encapsulates br_bks, serializedPlanTree, planTree, aes.

7 Multi-query Optimization Verification

We use spatial data set introduced in [9] to test the efficiency. It is a real data set that contains 104,770 California's Points of Interest. The original WGS 84 is projected into UTM zone 10N. The experiment is performed on an AMD Athlon 64 X2 Dual Core Processor 3800+ machine with 2GB RAM. The operating system is Ubuntu 9.10 with kernel 2.6. The file system is ext4, gcc version is 4.4.1, comiler optimization option is -O2, the PostGIS version is 1.4.

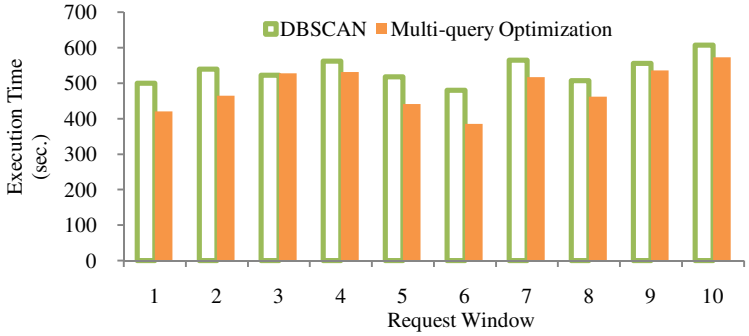


Fig. 4. 10-request Window Efficiency

The experiment is designed to verify window-based multi-query optimization result. Each window contains 10 recent (ε,minPts) requests. Here we explain how to generate this data sets. Firstly we choose 10 samples of ε with different minPts, then use function normrnd(ε, ε/2,1,10) in matlab to generate 10 normal distributed random numbers for each ε, these 10 normal random numbers have the same minPts, finally compose each 10-request window by randomly selecting one in these 10 normal random numbers of 10 sample ε. The 10-request based window overall execution time compared with DBSCAN is illustrated in Fig. 4. From this figure we see most windows have better efficiency than DBSCAN's except the third one. The third window has a little more time because one Expand execution has much more time with 186 than DBSCAN's 96 seconds. From this experiment we see that the overall

execution time of 10-request window have a high ratio of more than 0.8 of brute force algorithm DBSCAN, there are two reasons contributing it: n of AES_n in table _AES is 5 which is too small to store enough historical results for scheduling, hence many requests are scheduled using brute force algorithm DBSCAN, another reason is we just use a simple strategy for replacing working slot when there is no unoccupied one, which leads to seldom Split or Expand algorithm are scheduled.

8 Conclusion

In this paper, we propose a new SQL extension called *Cluster-by* to support in-database clustering for spatial DBMSs. We give the formal syntax and semantics of the proposed extension, and propose a novel framework to support multiquery optimization based on historical clustering results and statistics. Then we demonstrate the effectiveness of the framework by providing our example implementation of the clause in PostgreSQL database system.

Some on-going works that we plan to develop are listed: 1) To verify our framework generality, we also will design the Shrink, Expand and Shuffle algorithm for other classical clustering algorighms, 2) The cost model also needs to be defined more accurately.

Acknowledgements. This work is partially supported by the National Science Foundation under Grant No. IIS-1017926, 0844342, CNS- 0709285, 2009AA12Z220 and 2009AA12Z226. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Oracle Spatial Developer's Guide 11g (11.1) (2009)
2. Ester, M., Kriegel, H., Sander, J., Wimmer, M., Xu, X.: Incremental clustering for mining in a data warehousing environment. In: VLDB 1998, pp. 323–333 (1998)
3. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, pp. 226–231. AAAI Press (1996)
4. Li, F., Liu, S., et al.: An inheritable clustering algorithm suited for parameter changing. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics, vol. 2, pp. 198–203 (2004)
5. Frank, R., Jin, W., Ester, M.: Efficiently mining regional outliers in spatial data. In: Papadias, D., Zhang, D., Kollios, G. (eds.) SSTD 2007. LNCS, vol. 4605, pp. 112–129. Springer, Heidelberg (2007)
6. Guting, R.H.: An introduction to spatial database systems. VLDB Journal 4, 357–399 (1994)
7. Kalnis, P., Papadias, D.: Multi-query optimization for on-line analytical processing. Information Systems 278(5), 457–473 (2001)
8. Li, C., Wang, M., Lim, L., et al.: Supporting ranking and clustering as generalized order-by and group-by. In: SIGMOD 2007, pp. 127–138 (2007)

9. Li, F.-f., Cheng, D., Hadjieleftheriou, M., Kollios, G., Teng, S.-H.: On trip planning queries in spatial databases. In: Medeiros, C.B., Egenhofer, M., Bertino, E. (eds.) SSTD 2005. LNCS, vol. 3633, pp. 273–290. Springer, Heidelberg (2005)
10. Ordonez, C.: Integrating k-means clustering with a relational dbms using sql. IEEE Trans. on Knowl. and Data Eng. 18(2), 188–201 (2006)
11. Santos, M.Y., Moreira, A.: Automatic classification of location contexts with decision trees. In: CSMU 2006, pp. 79–88 (2006)
12. Shekhar, S., Chawla, S., Ravada, S., Fetterer, A., Liu, X., Lu, C.T.: Spatial databases: Accomplishments and research needs. IEEE Transactions on Knowledge and Data Engineering 11, 45–55 (1997)
13. Silva, Y.N., Aref, E.: Similarity group-by. In: ICDE 2009, pp. 904–915 (2009)
14. Yan, W., Larson, P.: Interchanging the order of grouping and join. In: Technical report (1995)
15. Yan, W.P., Larson, P.A.: Eager aggregation and lazy aggregation. In: VLDB 1995, pp. 345–357 (1995)
16. Zhang, C., Huang, Y.: Cluster by: a new sql extension for spatial data aggregation. In: ACM GIS 2007 (2007)

Multi-node Scheduling Algorithm Based on Clustering Analysis and Data Partitioning in Emergency Management Cloud

Qingchen Zhang, Zhikui Chen, and Liang Zhao

School of Software Technology, Dalian University of Technology, Dalian, China, 116620
{qingchen,matthew1988zhao}@mail.dlut.edu.cn, zkchen@dlut.edu.cn,

Abstract. Real-time processing is a key problem for big data analysis and processing, especially in emergency management. Strongly promoted by the leading industrial companies, cloud computing becomes increasingly popular tool for emergency management, that is emergency management cloud. How to make optimal deployment of emergency management cloud applications is a challenging research problem. The paper proposes a multi-node scheduling algorithm based on clustering analysis and data partitioning in emergency management cloud. First, the presented method divides the cloud nodes into clusters according to the communication cost between different nodes, and then selects a cluster for the big data analysis services. Second, the load balancing theory is used to dispatch big data analysis to these computing nodes in a way to enable synchronized completion at best-effort performance. At last, to improve the real-time of big data analysis, the paper presents a multi-node scheduling algorithm based on game theory to find optimal scheduling strategy for each scheduling node. Experimental results show the effectiveness of our scheduling algorithm for big data analytics in emergency management.

Keywords: Big data, multi-node scheduling, emergency management cloud.

1 Introduction

In recent years we have witnessed efforts to open up the Internet of Things and to make its development more inclusive [1]. Recently, more and more embedded devices are joined in IoT to monitor all kinds of objects, including traffic facilities, buildings, and lakes and so on, which makes the size of the data very huge [2]. In other word, we are living in an era where data is being generated from many different sources such as sensors, mobile devices and RFID [3]. To be specific, sensors distributed in different geographic locations collect data continuously from various objects before the amount of long accumulated data is extremely huge [4]. Besides, collected data from numerous mobile devices for multi-target tracking can exceed hundreds of terabytes and be continuously generated. Such big data represents data sets that can no longer be easily analyzed with traditional data management methods and infrastructures and pose a huge challenge on emergency management [5].

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 165–174, 2013.

Recent emergency situations in the world show the tendency that the occurrence frequency of natural disasters is expected to increase in future [6]. The effects of natural disasters are very serious and the destruction caused may take a very long time to recover. Real-time analysis and processing for big data is a key to improve emergency management.

The advent of Cloud Computing has been enabling enterprises to deal with such big data in time in emergency management, which is also called emergency management cloud, by leveraging vast amounts of computing resources available on demand with low resource usage cost [7]. With emergency management cloud, any one can share data and information with others over the Internet. What is more important, even if the data centre is ever affected by a natural disaster, data are safe there as well as there are contingency plans to transfer data to other centers if a disaster can be forecast. In addition, massive sensor data collected from various sensors can be processed and responded in real-time with emergency management cloud, especially facing sudden disasters. Therefore, emergency management cloud has become a hotspot in research in recent year. When deploying a cloud application in an emergency management cloud, the application user needs to select a number of cloud nodes including servers and virtual machines to run the cloud applications. How to make optimal deployment of emergency management cloud applications is a challenging and urgent required research problem.

In order to optimize a parallel data analysis and processing in cloud environment, this paper addresses: (a) node selection, i.e., "how many" and "which" computing nodes in cloud should be used, (b) data partition and synchronized completion, i.e., how to optimally apportion big data across parallelized computation environments to ensure synchronization, where synchronization refers to completing all workload portions at the same time even when resources and inter-networks are heterogeneous and situated in multiple Internet-separated clouds, and (c) multi-node scheduling, i.e., how to use multi-node scheduling to reduce the latency of waiting for many tasks.

To address these problems, we develop a novel multi-node scheduling algorithm for big data analysis and processing based on clustering analysis and data partitioning in emergency management cloud. Most of current methods usually rank the available cloud nodes based on their QoS values and select the best performing ones. A drawback of the ranking methods is that these methods cannot reflect the relations between different cloud nodes. So, our method considers not only the QoS ranking of nodes, but also the communication relations between them. In addition, the load balancing theory is used to dispatch big data analysis to different computing nodes in a way to enable synchronized completion at best-effort performance. Besides, current cloud computing scheduling strategy assumes that only one node is responsible for scheduling, which will increase latency of waiting for scheduling. In order to process many tasks in real time, multiply nodes are required to attend to the task scheduling process. The paper views the task scheduling of multiple nodes as a non-cooperative game and constructs a task scheduling model based on complete information static game. At last, we find the optimal scheduling strategies for each node by seeking Nash equilibrium, making the average completion time of each scheduling node minimum.

At last, we design a series of experiments to evaluate the performance of the presented algorithm. The experimental results show that our approach outperforms other existing methods for big data analytics in real time in emergency management cloud.

2 Related Work

QoS can be employed for describing the non-functional performance of cloud nodes [8]. Based on the cloud node QoS performance, a number of selection and schedule strategies have been proposed in the recent literature [9]. These previous methods just consider the order of the node performance, and not consider the relationship between nodes. In this paper, we focus on analyzing the relationship between cloud nodes to achieve optimal deployment of cloud applications. Some approaches have introduced task schedulers with load balancing techniques in cloud computing environments[10]. These methods have mainly focused on keeping the order of tasks in the queue while increasing performance by utilizing an external cloud on demand. However, they do not consider how many and which clouds are required and how much data is allocated to each chosen cloud for parallel processing. Similar with our approach, research efforts for task scheduling have been made to deploy parallel applications over massively distributed computing environments to analyze big data, such as MapReduce, Pregel and Dryad and so on[11-13]. Using only one node for scheduling may get high performance for data analytics if the single node has enough computational power. However, when big data arrive in the same time, this method will increase the latency for data analysis. In this paper, multiple nodes are required to attend to the task scheduling process to improve the real-time for big data processing.

3 Scheduling Algorithm Based on Clustering Analysis and Data Partitioning

There are a number of available distributed nodes in the cloud. Cloud user need to deploy their cloud applications on a number of optimal cloud nodes and use it. We divide the cloud nodes into clusters mainly based on clustering method, making the communication between nodes in the same cluster smallest.

Assume there are n cloud nodes distributed in a cloud, the response times between nodes can be represented as an n by n matrix, where p_{ij} is the response time between node i and node j. Apparently, this matrix is a symmetric matrix.

$$P = \begin{pmatrix} 0 & p_{12} & \cdots & p_{1n} \\ p_{21} & 0 & \cdots & p_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ p_{n1} & p_{n2} & \cdots & 0 \end{pmatrix} \tag{1}$$

A cluster analysis algorithm is designed to divide the cloud nodes into K clusters.They satisfy the following condition.

$$\begin{cases} C_i \neq \varnothing \quad i = 1, 2, ..., K \\ C_i \cap C_j = \varnothing \quad i, j = 1, 2, ..., K \ and \ i \neq j \\ \bigcup_{i=1}^{K} C_i = D \end{cases} \tag{2}$$

The following formula is used to calculate the distance D between a node to the centroid of the K-th clustering.

$$D = \lambda \times | cal_i - cal_{ck} | + (1 - \lambda) \times \frac{1}{d} \sum_{j: j \in C_k} p_{ij} \tag{3}$$

After the completion of the clustering, the nodes in the cloud platform are divided into several clusters. And then we select one of the clusters to perform computing tasks. There are many nodes in a cluster, so the algorithm intends to parallelize the big analysis task by dividing input data for multiple computing nodes in the same cluster. The paper partitions the data based on the load balancing so that each node has the same processing delay, including communication delay and calculate the delay, to avoid the reduction of the real-time because of the delay of single node.

If the average delay of the task for a unit of data on a node i is denoted as t_i and t_i includes two part, data transmission time and data processing time. And then the overall delay for executing a data size s_i (that is provided to node i for processing task) is $s_i t_i$ .In order to ensure ideal parallelization for n nodes and a set of data, the following formula is satisfied.

$$T = s_1 t_1 = s_2 t_2 = ... = s_n t_n \tag{4}$$

Let s be the total amount of the data s, we can get the following formula.

$$T = s / \sum_{i=1}^{n} 1 / t_i \tag{5}$$

From the above formulas, we can get the following formula.

$$s_i = s / t_i \sum_{i=1}^{n} 1 / t_i \tag{6}$$

Depending on these formulas, the paper can find the optimal data partitioning solution.

4 Multi-node Scheduling Model Based on Complete Information Static Game

After the data arrive, they need to be deployed into compute nodes in the cloud computing platform for parallel processing. Current cloud computing scheduling strategy assumes that only one node is responsible for scheduling, however, in order to process many tasks in real time, multiple nodes are required to attend to the job scheduling process. The paper views the job scheduling of multiple nodes in the cloud computing as a non-cooperative game and constructs a task scheduling model based on complete information static game. And then, we find the optimal scheduling strategies for each node by seeking Nash equilibrium.

We assume there are m nodes responsible for scheduling n nodes for task execution in the cloud computing platform. For each scheduling node *i*, the amount of data that need to be distributed once is *d* and we assume that the size of data the computing node *j* can process every time is *y*. We assume that the probability that the scheduling node *i* sends data to the computing node *j* is *p*.

The average completion time of the task of every scheduling node consists of the data transfer time and the processing time in the computing nodes. The average completion time that the scheduling node i dispatches the task to the computing node j can be expressed as the formula (7).

$$R_{ij}(p) = \omega_j(p) + t_{ij} \tag{7}$$

Where, *w(p)* denotes the processing time of the computing node *j*, determined by the processing capacity of the compute node *j*, and *t* represents the transfer time from the node *i* to the node *j*. We assume that average bandwidth available from the node *i* to the node *j* is *c*, each transfer time can be expressed as formula (8).

$$t_{ij} = p_{ij}\overline{d_i} / c_{ij} \tag{8}$$

Above all, The average completion time that the task scheduling the node *i* schedules the task to n compute nodes can be expressed as the formula (9).

$$R_i(p) = \sum_{j=1}^{n} p_{ij} R_{ij}(p) = \sum_{j=1}^{n} p_{ij}\omega_j(p) + \sum_{j=1}^{n} p_{ij}^2 d_i / c_{ij} \tag{9}$$

Definition 1. The scheduling game model *G* is defined a *G=(I,S,U)*, where *I=(1,2,…,i,…m)* represents the set of game makers, namely *m* scheduling node. The pure strategy space of every scheduling node *i* is $S_i=\{1,2,\ldots,j,\ldots,n\}$. The mixed strategy space of the node *i* is $\sum=\{p_i\}$, where $pi=\{p_{i1},p_{i2},\ldots p_{im}\}$ is one of the mixed strategies of

the node *i*. $U=\{u_1,u_2,\ldots,u_m\}$ is the income of the scheduling nodes. According to the formula (9), The income function of every node is

$$u_i(p)=\sum_{j=1}^{n} p_{ij}R_{ij}(p)=\sum_{j=1}^{n} p_{ij}\omega_j(p)+\sum_{j=1}^{n} p_{ij}^2 d_i / c_{ij} \quad .$$

Definition 2. The Nash equilibrium of *G* is defined as $p^*=\{p_1^*,p_2^*,\ldots,p_i^*,\ldots p_m^*\}$, where p_i^* is one of the mixed strategies of the node *i*, if and only if $u_i(p_i,p_{i-1}^*)<u_i(p_i^*,p_{i-1}^*)$, for every scheduling node *i*.

Definition2 can make sure that every scheduling node gets the largest income, namely the smallest average completion time of task, as the formula (10).

$$\min\{u_i(p)\},\quad \sum_{j=1}^{n} p_{ij}=1\ ;\ p_{ij}\geq 0\ ;\ \sum_{i=1}^{m} p_{ij}d_i \leq y_j \tag{10}$$

The presented game model calculates the best mixed strategies according to the definition 2 and the formula (10), making the average completion time of every node smallest and meeting the real-time request for processing big data.

5 Experiment

5.1 Experiment Setup

Our experimental environment consists of 16 distributed nodes as cloud computing nodes, each of which has a 2.8GHz core, 1GB memory and 250GB hard drive, and 3 nodes as scheduling nodes, each of which has four 3.2GHz cores, 4GB memory and 500GB hard drive. The data in experiments are collected from the digital home lab, including three sets of data: temperature, humidity, and carbon dioxide concentration, and the total size is up to 80GB. In order to evaluate the performance of our presented algorithm, we mainly run the outliers detection algorithm, which is important to emergency management, such as helping detect abnormal data and position sudden event location, in our cloud environment. To compare the performance of our approach against other methods, the metric that we use is makespan, which is defined as the duration between sending out a job and receiving a correct result.

5.2 Impact of Data Transfer Delay on Overall Execution Time

We first show the performance characteristics of computing nodes in the context of data computation and data transfer delay. Fig.1 shows the performance characteristics when we run the mining task with the entire data set.

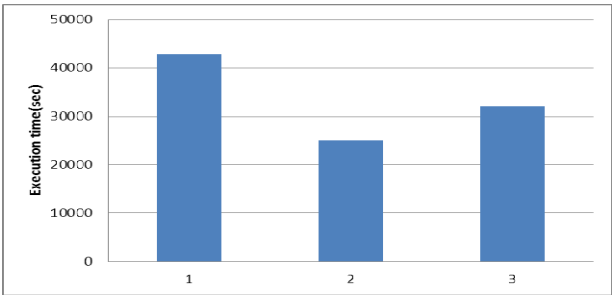


Fig. 1. Performance charateristics of computing nodes

In this figure, the first bar denotes the execution time of running in one virtual machine and the second bar represents that of running in 4 virtual machines, all of which are in the same computer, in parallel. The execution time of running of running in parallel in 4 virtual machines, however, any two of which are distributed in the different computers, is shown as the third bar. All of the machines have the same configuration. From this result, we can get two conclusions.

1) The outlier detection algorithm running in one virtual machine cost the longest time, while the execution time when using four virtual machines to run the algorithm in parallel is lower. This explains the need of parallel execution of big data analytics to improve the performance.

2) The execution time shown as the third bar is higher than that denoted by the second bar because of a significant delay for outlier detection of big data to get transferred over the different computing nodes. Therefore, we have to deal with the data transfer delay carefully.

5.3 Performance Comparison with a Single Scheduling Node

To study the performance of our presented method, we compare our method with the following two approaches.

Random-based: The scheduling algorithm with random-based cloud nodes selection.

QoS-based: The scheduling algorithm with ranking cloud nodes depending on the QoS of the nodes.

In this experiment, we partition cloud nodes in 3 clusters and use the parameter setting: $\lambda = 0.5$ and there is only one single scheduling node. The result shows that the execution time of all the algorithms increases as the size increases. Among all the methods, the scheduling algorithm with random-based nodes selection gets the worst performance and the result is not stable. Because of considering the QoS of cloud nodes during selecting cloud nodes for mining tasks, in most case, the performance of the QoS-based node scheduling algorithm is better than the random-based scheduling algorithm. However, when the size of data is smaller than 1.5GB, the execution time of the QoS-based node scheduling algorithm is lower than that of our method, because computing time occupies most of the overall execution time. With the increasing of

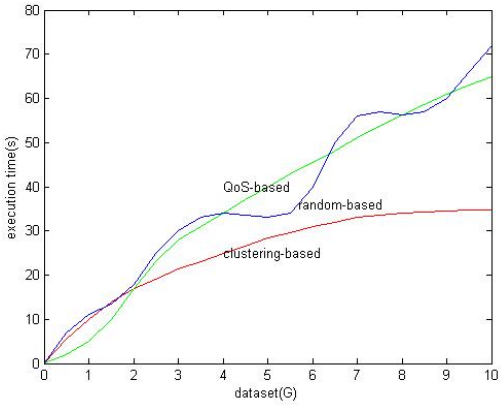


Fig. 2. Performance comparison of three algorithms

the size of data, especially when the size exceeds 2GB, the time of transfer and communication between different cloud nodes increases rapidly, our method obtains better performance than the QoS-based node scheduling algorithm which cannot consider the relations between cloud nodes. In conclusion, in most case, our method gets the best performance among all the scheduling approaches.

5.4 Performance Comparison with Multiple Scheduling Nodes

At last, in order to evaluate the performance of the presented multi-node scheduling algorithm based on game theory, we compare the multi-node scheduling algorithm with the scheduling method with single scheduling node. In this experiment, we use three scheduling nodes in the multi-node scheduling algorithm. The result is as shown in fig.3.

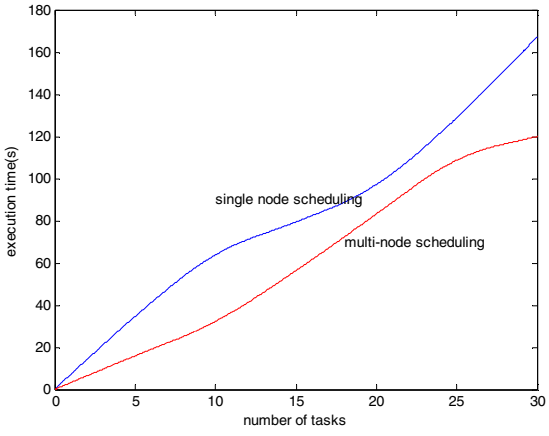


Fig. 3. Performance comparison of two scheduling algorithms

From fig.3, we can see that the execution time increases with the increasing of the number of tasks, but the multi-node scheduling algorithm has better performance than the scheduling algorithm with only one scheduling node. Because our method can schedule multiple tasks at the same time, reducing the waiting time of tasks.

6 Conclusion

Cloud computing is an effective tool for emergency management. With emergency management cloud, data can be processed in time and can be prevented from the sudden disasters. In this paper, we propose a novel multi-node scheduling algorithm to optimize the performance of big data analytics that can be run in distributed computing environment such as cloud computing platform. Generally speaking, our algorithm has supported decision makings on node selection, data partition and multi-node scheduling. We have compared our algorithm with other scheduling methods. Experiment shows that the performance of our algorithm is better than other approaches.

References

1. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer Networks 54(15), 2787–2805 (2010)
2. Kortuem, G., Kawsar, F., Fitton, D., et al.: Smart objects as building blocks for the internet of things. IEEE Internet Computing 14(1), 44–51 (2010)
3. Ashton, K.: That 'Internet of Things' Thing. RFiD Journal 22, 97–114 (2009)
4. Qin, X.-P., Wang, S.: Big Data Analysis—Competition and Symbiosis of RDBMS and MapReduce. Journal of Software 23(1), 32–45 (2012)
5. Cheng, Y., Qin, C., Rusu, F.: GLADE: big data analytics made easy. In: Proc. of the 28th International Conference on Management of Data, pp. 697–700 (2012)
6. Velev, D., Zlateva, P.: Principles of Cloud Computing Application in Emergency Management. In: Proc. of the International Conference on E-business, Management and Economics, pp. 119–123 (2011)
7. Iosup, A., Ostermann, S., Yigitbasi, M.N., et al.: Performance analysis of cloud com-puting services for many-tasks scientific computing. IEEE Trans. on Parallel and Distributed Systems 22(6), 931–945 (2011)
8. Zhang, Y., Huang, G., Liu, X.: Integrating resource consumption and allocation for infrastructure resources on-demand. In: Proc. of the 3rd IEEE International Conference on Cloud Computing, pp. 75–82 (2010)
9. Budati, K., Sonnek, J., Chandra, A.: Ridge: combining reliability and performance in open grid platforms. In: Proc. of the 16th International Symposium on High Performance Distributed Computing, pp. 55–64 (2007)
10. Frey, J., Tannenbaum, T., Livny, M.: Condor-G: A computation management agent for multi-institutional grids. Cluster Computing 5(3), 237–246 (2002)

11. Kim, H., Parashar, M.: CometCloud: An Autonomic Cloud Engine. Cloud Computing: Principles and Paradigms, 275–297 (2011)
12. Chen, Q., Hsu, M., Zeller, H.: Experience in Continuous analytics as a Service (CaaaS). In: Proc. of the 14th ACM International Conference on Extending Database Technology, pp. 509–514 (2011)
13. Huang, Y.C., Ho, Y.C., Lu, C.H., et al.: A cloud-based accessible architecture for large-scale adl analysis services. In: Proc. of the 4th IEEE International Conference on Cloud Computing, pp. 646–653 (2011)

Minimum-Delay POIs Coverage under Obstacle-Constraint in Emergency Management

Wenping Chen, Si Chen, and Deying Li

School of Information, Renmin University of China, Beijing 100872, P.R. China

Abstract. Applying wireless sensor network to emergency management is helpful to predict latent disaster and prevent or lessen the harm. In some cases of emergency management, it is not necessary to monitor the entire area all the time, and only need to employ a few of mobile sensors to monitor a number of Points of Interest ($POIs$) periodically. Due to the cost restriction, the number of mobile sensors is limited. Moreover, there may be some obstacles in the monitoring field, which makes mobile sensor cannot reach some $POIs$ directly. In this paper, we address the Minimum-Delay $POIs$ Coverage problem in Emergency Management, which is how to schedule the limited number of mobile sensors to monitor the $POIs$ in a region with obstacles such that the $POIs$ coverage delay is minimized. Firstly, we calculate the shortest distance between any two $POIs$ in the monitoring field with obstacles to construct a weight complete graph. Secondly, we propose an algorithm named $Obstacle-TSP-S$ to address the minimum-delay POIs coverage problem. By the comprehensive simulations, we evaluate the performance of the proposed algorithm. The simulation results show the efficiency of our algorithm.

Keywords: emergency management, mobile wireless sensor networks, obstacle, cost-constraint, $POIs$ coverage, minimal delay.

1 Introduction

Wireless Sensor Networks ($WSNs$) are regarded as a promising tool for monitoring the physical world. Applying $WSNs$ to emergency management is helpful for predicting latent disaster and finding the danger in good time, which can prevent or lessen the harm. In some emergency management cases such as monitoring mine exploitation and dangerous substance leakage, it is not necessary to monitor the entire area all the time, only a number of critical points called Points of Interest ($POIs$) need to be monitored periodically. Continuous coverage in such cases is undoubtedly wasteful. Therefore, the sweep coverage which employs the mobile sensors to monitor the $POIs$ at regular intervals is more efficient in emergency management.

The existing works [2],[3] about sweep coverage problem focus on how to schedule minimum number of mobile sensors in obstacle-free area to sweep all

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 175–186, 2013.

$POIs$ within specified sweep period. They assumed that there are enough mobile sensors for monitoring $POIs$. However, in real applications, there may have restriction on the number of mobile sensors due to the system cost concern. Furthermore, there may be some obstacles in the monitoring field, where mobile sensors cannot reach some $POIs$ alone straight line direction.

In this paper, we address the Minimum-Delay $POIs$ Coverage problem in emergency management with two constraints, obstacle-constraint and cost-constraint. Our target is how to schedule the limited number of mobile sensors monitoring in a region with obstacles to minimize the delay of $POIs$ coverage. As shown in Fig.1, there are three mobile sensors in the area. Each mobile sensor sweeps specific $POIs$ along its trajectory respectively. Since there are a few of obstacles in this area, the mobile sensors probably cannot reach a POI from another POI directly. The delay of $POIs$ coverage is defined as the time interval from sensors starting sweeping to all the sensors finishing their monitoring task. We assume the moving speed of all mobile sensors is same. Thus, the delay of $POIs$ coverage is decided by the longest sweeping trajectory, which is M1's.

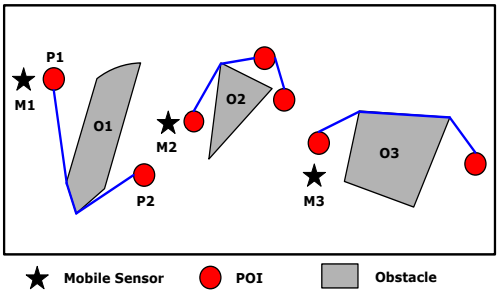


Fig. 1. The Network Model of the Detection Region

To address this problem, we first calculate the shortest distance between any two $POIs$ in the area with obstacles. Second, we design the trajectory for each mobile sensor. We take the $POIs$ and the shortest distance between any two $POIs$ as the input of Travel Salesman Problem (TSP), and construct a TSP ring by using the algorithm for TSP, then divide the ring into k trajectories and let every mobile sensor move along one of the trajectories to sweep the $POIs$ on the trajectory.

The rest of this paper is organized as follows. Section 2 reviews the existing works about coverage problem in $WSNs$. In section 3 we present the network model and the definition of Minimum-Delay $POIs$ Coverage with Obstacles in emergency management. The $Obstacle - TSP - S$ algorithm is proposed in Section 4. Then, we evaluate the performance of our algorithm through extensive simulations in Section 5.

2 Related Work

Many efforts have been made on full coverage problem (including point coverage ([4]-[8]) and area coverage([9]-[18]) and barrier coverage problem ([19]-[26])in $WSNs$. Full coverage and barrier coverage often require that target or area is covered all the time. To achieve full coverage or barrier coverage, a majority of works studied the static coverage under the requirement of continuous coverage. Although it can achieve the best coverage, the system cost is prohibitive. To improve the coverage quality efficiently, a type of mixed network infrastructure called hybrid network is adopted in some applications, which is composed of mobile sensors as well as static sensors ([27]-[30]).

The sweep coverage problem is motivated by the applications without continuous coverage requirement. In such cases, mobile sensors are often used. There are some works about sweep coverage [2],[3]. The main goal of those works is to minimize the number of mobile sensors so as to cover all the $POIs$ in a region without violating the given time. Li et al. proposed a centralized algorithm CSWEEP [2]. Firstly, take all $POIs$ as input and employ a PTAS algorithm [31] for TSP and get an approximate TSP ring. Secondly, schedule every mobile sensor move along the TSP ring segments back and forth under the requirement of sweep period. Du et al. [3] proposed two algorithms for different situations. In the first algorithm, they gradually deploy mobile sensors and schedule the mobile sensors to move on the same trajectory in each period. In the second algorithm, OSweep, the mobile sensors are not required to follow the same trajectory in each period, which schedule all the mobile sensors to move along the TSP ring which consists of $POIs$ towards the same direction. The OSweep algorithm outperform CSWEEP algorithm.

The above works on sweep coverage assumed that they can obtain enough mobile sensors. However, in real applications, due to the cost constraints, there may be not enough mobile sensors to accomplish monitoring. Furthermore, both of them assume there are not obstacles in monitoring region. In this paper, we take cost and obstacle restriction into account to address the minimum-delay POIs Coverage problem in emergency management.

3 Minimum-Delay POIs Coverage under Obstacle-Constraint

3.1 Network Model

Given a set of mobile sensors $M = \{m_1, \ldots, m_k\}$ to monitor the set of $POIs$ $P = \{p_1, \ldots, p_m\}$ in a two-dimensional region R. We assume each POI has a globally unique ID and a fixed position. In region R, there are some obstacles, such as buildings, mountains and ponds, which maybe block mobile sensors to move. Suppose there are n obstacles in the region, saying in set $O = \{o_1, \ldots, o_n\}$. Let $d(u, v)$ be the shortest distance between any two points u and v in the region. We have the following assumptions. Firstly, the start positions of mobile

sensors are not determined in advance. Secondly, all the mobile sensors move at a constant speed v in the region. Thirdly, the sense radius of mobile sensors is very small that mobile sensors must pass the locations of $POIs$ to monitor the $POIs$.

In this paper, we study the minimum-delay $POIs$ coverage under obstacle-constraint. Based on the precondition that the position and shape of each obstacle is known, we aim to schedule k mobile sensors to moniter specific $POIs$ in their own trajectories such that the delay to cover the $POIs$ is minimized.

3.2 Problem Definition

Before formalizing the *Minimum-Delay POIs Coverage problem under Obstacle-Constraint in Mobile Wireless Sensor Networks* problem, we first introduce several definitions as follows.

Definition 1. *The Delay of POIs Coverage: The delay of POIs Coverage is the duration from the mobile sensors starting to sweep to the last POI having been scanned.*

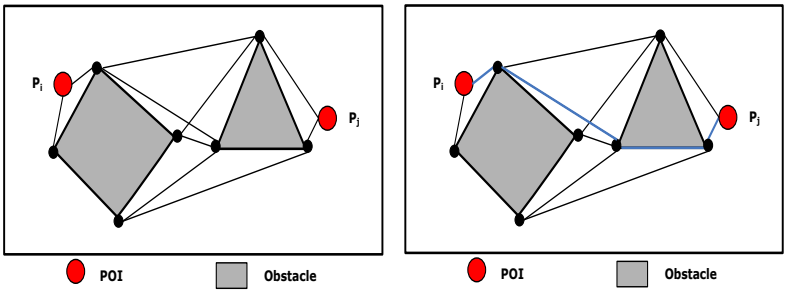
Definition 2. *Minimum-Delay POIs Coverage problem under Obstacle-Constraint: Given k mobile sensors and the deployments of m POIs, the Minimum-Delay POIs Coverage under Obstacle-Constraint problem is how to schedule k mobile sensors to move in a region with obstacles such that the delay of the POIs coverage is minimized.*

4 Algorithm for the Minimum-Delay POIs Coverage Problem under Obstacle-Constraint

In this section, by formulating the Minimum-Delay POIs Coverage problem under Obstacle-Constraint into a TSP, we propose the *TSP-S-Obstacle* algorithm to resolve the problem.

If there is no restriction on the start positions of mobile sensors, given the deployment of $POIs$, we can generate an undirected complete graph $G = (V, E, W)$, where V is the set of all $POIs$. For any two $POIs$ p_i, p_j, there is an edge e_{ij} between p_i and p_j(i.e., $e_{ij} \in E$), and $w(e_{ij}) = d(p_i, p_j)$, $d(p_i, p_j)$ is the shortest distance between p_i and p_j.

The main idea of *TSP-S-Obstacle* algorithm is as follows. We first calculate the shortest distance between each pair of $POIs$ under obstacle-constraint. Secondly, take the weighted completed graph G as the input of TSP, and use the approximate algorithm [31] to obtain a TSP ring. Finally, we generate k trajectories from the TSP ring for the k mobile sensors, which the maximum length among all k trajectories is minimized. The details are discussed in the following sections.



(a) Constructing visibility graph (b) Finding shortest path by Dijkstra algorithm

Fig. 2. Finding the shortest path under obstacle-constraint

4.1 Finding the Shortest Path under Obstacle-Constraint

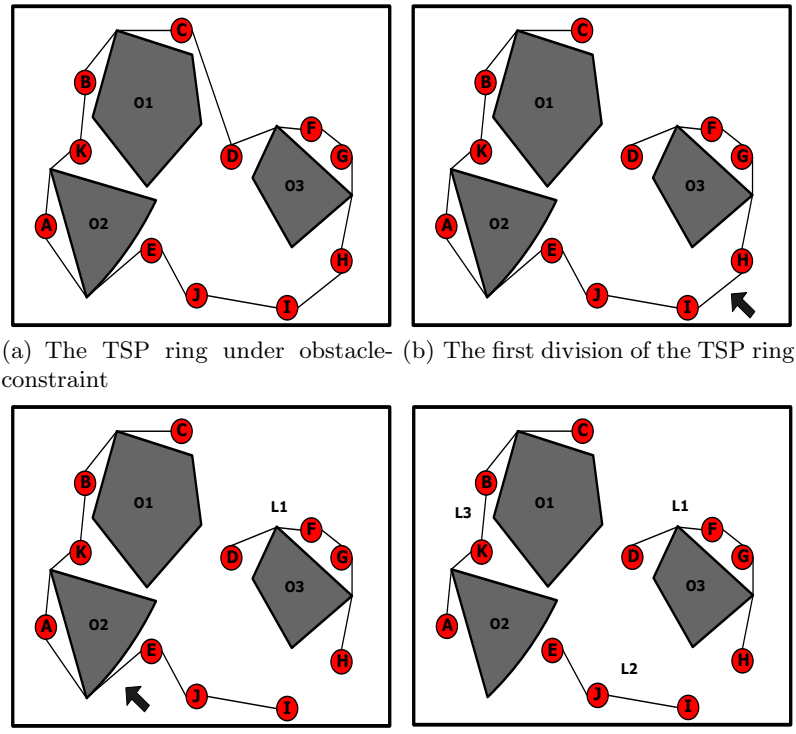
In this subsection, we briefly introduce the method to find the shortest path between any two $POIs$ under obstacle-constraint[1]. We assume that each obstacle is a polygon and its shape and position is known in advance. The main idea of the method is constructing a visibility graph and employing $Dijkstra$ algorithm to find the shortest path in the graph.

The shortest path between two points should follow straight line segments except when the straight line meet the the obstacles. As shown in Fig.2(a), there is a set of two disjoint obstacles O and two $POIs$ in the region R. $V(R)$ includes the the vertices of the obstacles O and two $POIs$. Let $G_{vis}(R) = (V(R), E_{vis})$ be the visibility graph of region R, where $E_{vis} = \{\overline{uv} | u, v \in V(R), \overline{uv} \subset \{R \setminus \cup O\}\}$. Based on the visibility graph of region R, we use $Dijkstra$ algorithm on $G_{vis}(R)$ to find the shortest path between two $POIs$ as shown in Fig.2(b).

4.2 TSP Based Searching Algorithm for the Minimum-Delay POIs Coverage

In this subsection, we propose the $TSP - S - Obstacle$ algorithm for the Minimum-Delay POIs Coverage with obstacles restriction. We aim to find k different trajectories passing by all $POIs$, where k is the number of mobile sensors, and the maximum length among all k trajectories is minimized.

After obtain the shortest path between any two $POIs$, we first create a weighted completed graph G. Then, we employ PTAS algorithm for TSP on the graph G, and get an approximate TSP ring L. Next, we delete the longest edge l_m in TSP ring and denote the current length of the TSP ring as L_{cur}, where $|L_{cur}|=|L|-|l_m|$. Then we could obtain an upper bound of trajectory length of mobile sensor $l_b = |L_{cur}|/k$. After that, choose one of the $POIs$ connected to the longest edge l_m in the TSP ring, and start from it to generate the trajectory t_1 by adding the $POIs$ in the TSP ring in sequence until the length of t_1 is larger than l_b. Then, we remove the latest added POI p_i from t_1 and get the first trajectory t_1. Next, delete the edge L_{del} which is between p_i and the



(a) The TSP ring under obstacle-constraint

(b) The first division of the TSP ring

(c) The second division of the TSP ring

(d) The result of POIs coverage under obstacle-constraint

Fig. 3. An example of POIs coverage for TSP-S-Obstacle

last POI in t_1. Meanwhile, we recalculate the bound l_b and let p_i be the start position of establishing next trajectory. The above steps are executed repeatedly until the generated trajectories contain all $POIs$.

Since every time a trajectory is generated,the upper bound of trajectory length l_b is updated to a smaller value, it is possible that the number of generated trajectories is less than the number of mobile sensors k but all $POIs$ have been covered. Once such situation appears, we have to adjust the trajectories. We will select the longest trajectory and split it into two parts evenly until there are k trajectories. The pseudocode of the $TSP - S - Obstacle$ algorithm is presented in $Algorithm1$.

Fig.3 illustrates the process of generating trajectories for mobile sensors by $TSP - S - Obstacle$ algorithm. There are three mobile sensors, a number of $POIs$ and three obstacles in the region. As shown in Fig.3(a), a TSP ring under obstacle-constraint has been obtained. In Fig.3(b),we remove the longest edge $\overline{CD}$ in the ring and calculated the bound l_b. Then we choose POI D as the start position and add $POIs$ F, G, H and I in sequence, when adding POI I to the current trajectory,the length of trajectory is larger than the bound l_b. Thus, as

illustrated in Fig.3(c), we delete $POI\ I$ from the current trajectory to generate the first trajectory $L1 = \{D, F, G, H\}$. Then we recalculate the bound l_b again and start from $POI\ I$ to generate the next trajectory. The second removed edge is $\overline{EA}$. Finally, three trajectories are generated, the new generated trajectories are $L2 = \{I, J, E\}$ and $L3 = \{A, K, B, C\}$ as shown in Fig.3(d). The three mobile sensors will move along the three trajectories respectively.

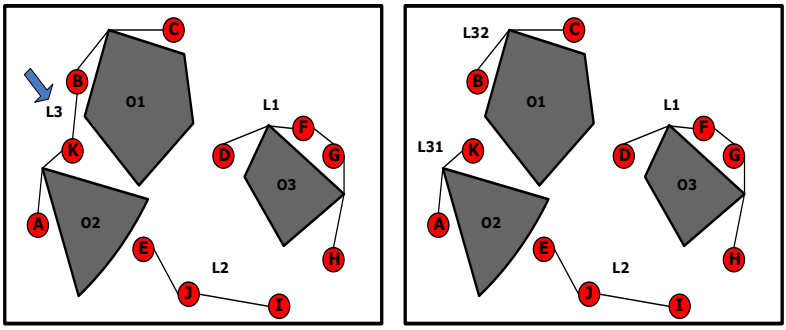
Algorithm 1. TSP-S-Obstacle(V, E, W, k, &c)

```
1: Construct an undirected weighted complete graph G in which the weight of each
   edge is the shortest distances between the corresponding two POIs under the
   obstacles-constraint;
2: Find a TSP ring L by using Traveling Salesman Problem approximation algorithm;
3: Delete the longest edge in L and set the right node as the current node V_cur, which
   is the start position of the first trajectory;
4: for (Each POIs in clockwise order of L) do
5:    if |l_tcur| + |v_cur, v_next| <= |l_cur|/k then
6:       Add v_next to the current trajectory;
7:       l_tcur ← l_tcur + |v_cur, v_next|;
8:    else
9:       Create a new trajectory with start node v_next;
10:      l_cur ← |l_cur| − |v_cur, v_next|;
11:      k − −;
12:   end if
13: end for
14: for (i = t_cur to k) do
15:    Find the longest trajectory and split it into two parts whose lengths are approx-
       imately equal;
16: end for
```

Another special case for $TSP - S - Obstacle$ algorithm is shown in Fig.4. There are four mobile sensors. However, after generating the third trajectory, all $POIs$ have been covered. To deal with such situation, we adjust the current trajectories as shown in Fig.4(a). We select the longest trajectory $L3$, and split it into two parts $L31$ and $L32$. After the adjustment, each mobile sensor takes charge of one trajectory as shown in Fig.4(b).

Theorem 1. *The time complexity of* TSP-S-Obstacle *algorithm is* $O(m^2h^2logh)$*, where* m *is the number of the POIs,* h *is the total number of edges of obstacles*

Proof. Firstly, since calculating the shortest distance between any two $POIs$ costs $O(h^2logh)$ [1] where h is total number of edges of all Obstacles, then constructing a weighted complete graph costs $O(m^2h^2logh)$. Secondly, creating an approximate TSP ring costs $O(m^2)$. And it takes $O(m)$ to divide and adjust the TSP ring into k trajectories. Therefore, the computational complexity of *TSP-S-Obstacle* algorithm is $O(m^2h^2logh)$. The proof finishes.



(a) Splitting the longest trajectory (b) The result of the special case

Fig. 4. A special case of POIs coverage for TSP-S-Obstacle

5 Performance Evaluation

we conduct extensive simulations to study the performance of our proposed algorithm. Since the length of each sensor's trajectory is various, the sweeping time is different from each other. Although one mobile sensor accomplishing the sweeping, it cannot start the next round of monitoring immediately until all sensors finish sweeping. Therefore, we study two important metrics: the delay and the ΔT of $POIs$ coverage schemes under obstacle-constraint, where ΔT is defined as the time interval from the first sensor accomplishing the sweeping to the last sensor finishing its own sweeping, which shows the maximal difference between the length of trajectory.

5.1 Simulation Setup

The situation of the monitoring region is shown in Fig.5. There are eight obstacles in a $3000m*3000m$ square area. The shapes and locations of all obstacles are known in advance. Note that when the area of obstacle is big enough, the shape of obstacle can be ignored. In our simulation, the shape of obstacles is rectangle or pentagon. All $POIs$ are randomly deployed in the region. Let S denote the number of mobile sensors. The moving velocity v of each mobile sensor is same, which is $80m/min$.

Since none of the previous works about sweep coverage considered the restrictions both on the obstacles and the number of mobile sensors, we compare our $TSP-S-Obstacle$ algorithm with the nearest neighbor first algorithm ($NNF-Obstacle$) which is a greedy scheme. In $NNF-Obstacle$ algorithm, each time the nearest neighboring POI of each mobile sensor is selected to add to the corresponding trajectory. Furthermore, we test the performance of $TSP-S-Obstacle$ algorithm under the condition that no obstacles in this region, which is a special case of $TSP-S-Obstacle$ algorithm. For the convenience of representation, when $TSP-S-Obstacle$ algorithm apply to the environment without obstacles, we call it $TSP-S$ algorithm.

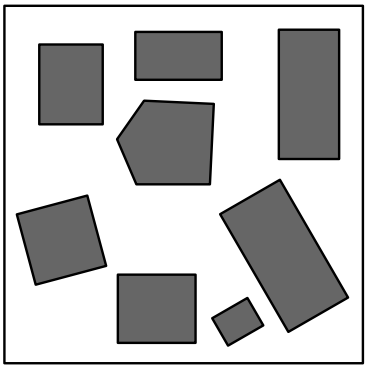
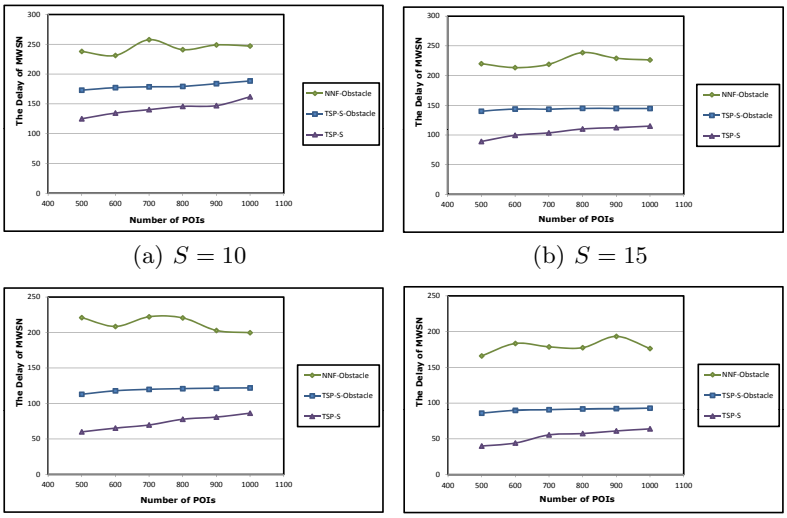


Fig. 5. The Simulation Map of the Detection Region



(a) $S = 10$ (b) $S = 15$

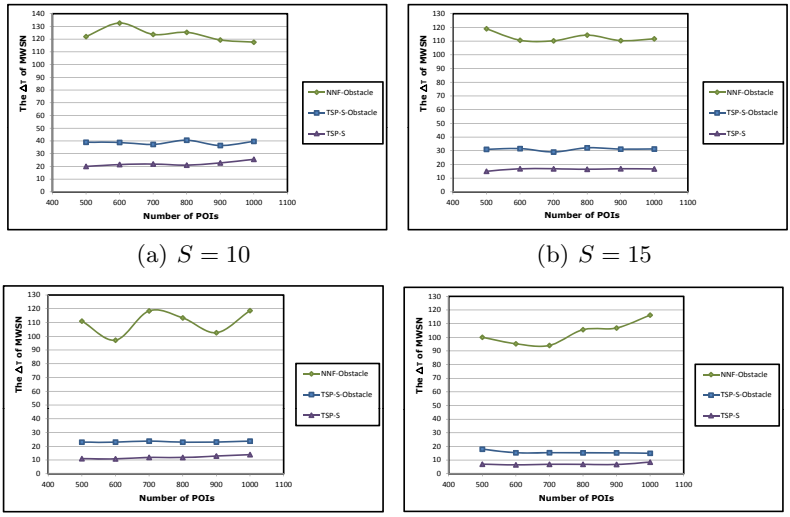
(c) $S = 20$ (d) $S = 25$

Fig. 6. The Performance for the Delay of Network

5.2 The Performance of TSP Based Searching Algorithm under Obstacle-Constraint

To evaluate the performance of $TSP - S - Obstacle$ algorithm, we change the number of $POIs$ and measure the delay and ΔT with given number of mobile sensors, the simulation results are shown in Fig.6 and Fig.7 respectively.

Firstly, we can find that the $TSP-S-Obstacle$ and $TSP-S$ algorithm always outperform the $NNF - Obstacle$ algorithm. It because that $NNF - Obstacle$ only guarantees that the number of $POIs$ in every trajectory of mobile sensor is



(a) $S = 10$ (b) $S = 15$

(c) $S = 20$ (d) $S = 25$

Fig. 7. The Performance of the ΔT

almost equal and can not ensure that the length of different trajectories of mobile sensors is approximately equal, whereas the $TSP - S - Obstacle$ and $TSP - S$ algorithm consider the global condition to generate the sweep trajectories. Moreover, it is obvious that the $TSP - S$ algorithm outperforms $TSP - S - Obstacle$ algorithm. It is because with the obstacles in the monitoring area, the shortest distance between each pair of $POIs$ increases compared to the scenario without obstacles.

Secondly, as shown in Fig.6,when the number of $POIs$ increasing,the distribution of $POIs$ is more compact and the obstacles restriction will have less effect on sweep trajectory generation. Thus,we can observe that the difference of delay between $TSP - S - Obstacle$ and $TSP - S$ decreases.

Thirdly, we can observe that increasing the number of mobile sensors can improve the performance of $TSP - S - Obstacle$ and $TSP - S$ algorithm significantly. This is mainly because that with more mobile sensors sweeping there will be less $POIs$ assigned to each mobile sensor. However, the performance $NNF - Obstacle$ is not always improved since the performance of $NNF - Obstacle$ largely depends on the deployment of the obstacles and $POIs$.

Furthermore, as shown in Fig.7, we observe that when the number of $POIs$ increasing, there is little difference of ΔT between the $TSP - S - Obstacle$ and $TSP - S$. Since the size of area is fixed, with the number of $POIs$ increasing, the average distances between $POIs$ decrease. Therefore, the fluctuation of ΔT is slight.

Acknowledgment. This research was supported in part by the National Natural Science Foundation of China under grants 61070191 and 91124001, and the National Research Foundation for the Doctoral Program of Higher Education of China (Grant 20100004110001).

References

1. de Berg, M., Cheong, O., van Kreveld, M., Overmars, M.: Computational Geometry. Springer (2000)
2. Li, M., Cheng, W., Liu, K., He, Y., Liu, Y., Liao, X.: Sweep Coverage with Mobile Sensors. IEEE Trans. Mobile Computing 10(11), 1534–1545 (2011)
3. Du, J., Li, Y., Liu, H., Sha, K.: On Sweep Coverage with Minimum Mobile Sensors. In: InternationalConference on Parallel and Distributed Systems, pp. 283–290 (2010)
4. Cardei, M., Du, D.Z.: Improving Wireless Sensor Network Lifetime through Power Aware Organization. ACM Wireless Networks 11(3), 333–340 (2005)
5. Cardei, M., Thai, M.T., Li, Y., Wu, W.: Energy-Efficient Target Coverage in Wireless Sensor Networks. In: IEEE INFOCOM, pp. 1976–1984 (2005)
6. Liu, H., Chen, W., Ma, H., Li, D.: Energy-Efficient Algorithm for the Target Q-coverage Problem in Wireless Sensor Networks. In: Pandurangan, G., Anil Kumar, V.S., Ming, G., Liu, Y., Li, Y. (eds.) WASA 2010. LNCS, vol. 6221, pp. 21–25. Springer, Heidelberg (2010)
7. Liu, H., Wan, P., Yi, C., Jia, X., Makki, S., Niki, P.: Maximal Lifetime Scheduling in Sensor Surveillance Networks. In: IEEE INFOCOM, pp. 2482–2491 (2005)
8. Chaudhary, M., Pujari, A.K.: Q-Coverage Problem in Wireless Sensor Networks. In: Garg, V., Wattenhofer, R., Kothapalli, K. (eds.) ICDCN 2009. LNCS, vol. 5408, pp. 325–330. Springer, Heidelberg (2008)
9. Cardei, M., MacCallum, D., Cheng, X., Min, M., Jia, X., Li, D., Du, D.Z.: Wireless Sensor Networks with Energy Efficient Organization. Journal of Interconnection Networks 3(3-4), 213–229 (2002)
10. Slijepcevic, S., Potkonjak, M.: Power Efficient Organization of Wireless Sensor Networks. In: IEEE ICC, pp. 472–476 (2001)
11. Bai, X., Xuan, D., Yun, Z., Lai, T.H., Jia, W.: Complete Optimal Deployment Patterns for Full-Coverage and K-Connectivity ($k \leq 6$) Wireless Sensor Networks. In: Proc. 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 401–410 (2008)
12. Kumar, S., Lai, T.H., Balogh, J.: On K-Coverage in a Mostly Sleeping Sensor Network. In: Proc. 10th Annual International Conference on Mobile Computing and Networking, pp. 144–158 (2004)
13. Wang, X., Xing, G., Zhang, Y., Lu, C., Pless, R., Gill, C.: Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks. In: Proc. ACM SenSys, pp. 28–39 (2003)
14. Liu, B., Brass, P., Dousse, O.: Mobility Improves Coverage of Sensor Networks. In: Proc. ACM MobiHoc, pp. 300–308 (2005)
15. Zhou, Z., Das, S., Gupta, H.: Connected K-Coverage Problem in Sensor Networks. In: International Conference on Computer Communication Networks, pp. 373–378 (2004)
16. Hefeeda, M., Bagheri, M.: Randomized k-Coverage Algorithms For Dense Sensor Networks. In: IEEE INFOCOM, pp. 2376–2380 (2007)

17. Wang, Y., Cao, G.: On Full-View Coverage in Camera Sensor Networks. In: IEEE INFOCOM, pp. 1781–1789 (2011)
18. Bai, X., Kumar, S., Yun, Z., Xuan, D., Lai, T.H.: Deploying Wireless Sensors to Achieve Both Coverage and Connectivity. In: Proc. ACM MobiHoc, pp. 131–142 (2006)
19. Kumar, S., Lai, T.H., Arora, A.: Barrier Coverage with Wireless Sensors. In: Proc. ACM MobiCom, pp. 284–298 (2005)
20. Chen, A., Kumar, S., Lai, T.H.: Designing Localized Algorithms for Barrier Coverage. In: Proc. ACM MobiCom, pp. 63–74 (2007)
21. Balister, P., Bollobas, B., Sarkar, A., Kumar, S.: Reliable Density Estimates for Coverage and Connectivity in Thin Strips of Finite Length. In: Proc. ACM MobiCom, pp. 75–86 (2007)
22. Yang, H., Li, D., Zhu, Q., Chen, W., Hong, Y.: Minimum Energy Cost k-barrier Coverage in Wireless Sensor Networks. In: Pandurangan, G., Anil Kumar, V.S., Ming, G., Liu, Y., Li, Y. (eds.) WASA 2010. LNCS, vol. 6221, pp. 80–89. Springer, Heidelberg (2010)
23. Ssu, K.F., Wang, W.T., Wu, F.K., Wu, T.T.: K-Barrier Coverage with a Directional Sensing Model. International Journal on Smart Sensing and Intelligent Systems 2(1), 75–93 (2009)
24. Saipulla, A., Westphal, C., Liu, B., Wang, J.: Barrier Coverage of Line-Based Deployed Wireless Sensor Networks. In: IEEE INFOCOM, pp. 127–135 (2009)
25. Liu, B., Dousse, O., Wang, J., Saipulla, A.: Strong barrier coverage of wireless sensor networks. In: ACM MobiHoc, pp. 411–420 (2008)
26. He, S., Chen, J., Li, X.: Cost-Effective Barrier Coverage by Mobile Sensor Networks. In: IEEE INFOCOM, pp. 819–827 (2012)
27. Wang, W., Srinivasan, V., Chua, K.C.: Trade-Offs Between Mobility and Density for Coverage in Wireless Sensor Networks. In: Proc. ACM MobiCom, pp. 39–50 (2007)
28. Wang, D., Liu, J., Zhang, Q.: Probabilistic Field Coverage using a Hybrid Network of Static and Mobile Sensors. In: Proc. IEEE IWQoS, pp. 56–64 (2007)
29. Ekici, E., Gu, Y., Bodag, D.: Mobility-Based Communication in Wireless Sensor Networks. IEEE Communications Magazine 44(7), 56–62 (2006)
30. Chellappan, S., Gu, W., Bai, X., Xuan, D., Ma, B., Zhang, K.: Deploying Wireless Sensor Networks under Limited Mobility Constraints. IEEE Trans. Mobile Computing 6(10), 1142–1157 (2007)
31. Du Ker-I Ko, D.Z., Hu, X.D.: Design and Analysis of Approximation Algorithms. Springer (2012)

A cloud Computation Architecture for Unconventional Emergency Management

Jianhui Li[1], Yuanchun Zhou[1,*], Wei Shang[2], Cungen Cao[3], Zhihong Shen[1], Fenglei Yang[1], Xiao Xiao[1], and Danhuai Guo[1]

[1] Computer Network Information Center, Chinese Academy of Sciences, Beijing, China
[2] Academy of Mathematics and System Science, Chinese Academy of Sciences, Beijing, China
[3] Institute of Computing Technology, Chinese Academy of Sciences,
Chinese Academy of Sciences, Beijing, China
{lijh,zyc,bluejoe,flyang,xiaoxiao,guodanhuai}@cnic.cn,
shangwei@amss.ac.cn,
cgcao@ict.ac.cn

Abstract. With the development of technologies and the deterioration of natural environment, unconventional emergencies outbreak more unexpectedly and diffuse more quickly and broadly. Secondary and derived disasters increase, and the impacts tend to be indirect and tremendous. Emergency management decisions are facing great challenges, and have attracted great concerns from government departments, academia and industries. In recent years, as a service-oriented computing mode, the cloud computing technology brings advantage in information sharing, resource allocating, and distributed high-performance computing, which makes it a feasible solution to unconventional emergency management, research, quick response and decision support. In this paper, we propose a cloud computation architecture for unconventional emergency management, which involves the key technologies including computation resource pooling, scalable extension of computation resource and services and user-centroid service management. The proposed architecture supports multilevel demand in computation and storage resource by providing services such as virtual machine, big data storage, web information detection and spatio-temporal data visualization. Three experimental scenarios are designed to validate the improvement of decision support capabilities and emergency response speed.

Keywords: Unconventional emergency, Cloud computing, Emergency management.

1 Introduction

Unconventional emergencies are the disasters which lack sufficient omen, but produce potential secondary and derived harm to the physical world and human society. Typical unconventional emergencies include natural disasters, production safety accidents, public health incidents and social security events. Although there are some

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 187–198, 2013.

indication and observable warning signals in these events, the outbreak and spread of most of unconventional emergencies cannot be predicted accurately and in time in advance by current technologies. Moreover, unconventional emergency usually takes place in a burst and evolves in no time so as to make it difficult to be solved in real time. Mass heterogeneous information is produced in the evolution. And those information are involved with different specific domains and distributed in different geographical location so it is possible to be solved after participation of the government and the society[1]. The current conventional countermeasures hardly provide necessary comprehensive support in preface detection, effective finding, early warning and quick response.

In recent years, as a service-oriented computing mode, the advantage of cloud computing technology in information sharing, resource allocation, and high-performance distributed computation makes it a feasible solution to build unconventional emergency management service platform[2]. In this paper, we propose emergency cloud architecture oriented to unconventional emergency management. It includes computing resource pooling technology that can achieve the integration and sharing of resources of all kinds of emergency management in a standard interface; rapid and elastic expansion of resources and services on demand that can allocate resources rapidly to provide services in the effective time window when unconventional emergencies occur and meanwhile ensure that the services and resources can be dynamically released when required less in normal times. Cloud computing can support all kinds of emergency management users to self-order and customize services on demand, and even rapidly assembly and develop new services, which will greatly improve the speed of the decision support capabilities and emergency response.

2 Related Works

Cloud computing is a new, scalable, demand-based computing model. It forms shared resource pool (including network, storage, servers, software and etc.) that can be accessed via Internet, and provide demand-based and measurable services for large amount of users. As bright prospect being showed, the cloud computing has a wide range of application scenarios in medical treatment and medicine, manufacturing, finance and energy, e-government, education and research, telecommunication, defense industry. In addition, cloud computing satisfies the requirements of integration, reliability and contractibility needed by conventional emergency management platform. However, its application in unconventional emergency management has just started.

Cloud computing is a kind of data-intensive supercomputing and has key technologies in the areas of data storage, programming mode and data management. It includes but not limited to parallel programming mode, mass data parallel storage technology, mass data management technology, virtualization technology, cloud computing platform management technology, and green energy-saving technology.

The current cloud computing research looking into unconventional emergency management mainly focused on how to control the development of specific events and what measures to take for disaster mitigation[3]. According to the research

problem perspective, it can be divided into: event characteristics and pre-warning grading [4, 5], event breaking and conduction mechanism [6, 7], generation and management of emergency plans [8], emergency command and coordination mechanism, parallel system based unconventional emergency computing experimental platform [9, 10]. With increasing number of various kinds of emergencies, the social impacts therefrom lead researchers to pay more attention to public health and social population emergency management [11] and management of public opinions about emergency[12]. Although the suggestions on developing emergency management platform had been proposed early[13], there are no practical technological solutions.

The unpredictability and high-complexity in the innovation of unconventional emergencies make the management platform more challenged. American Upp Technology enterprise puts forward a public health security emergency management system based on cloud computing, that is IRMS 360 Enterprise. The project provides enterprises with a solution plan: Data Center, Enterprise Integration, Business Intelligence and Enterprise Mobility. American ESRI enterprise offers various services and applications based on cloud to ensure effective and stable geographic information system (GIS) service for enterprises' emergency management.

As for cloud processing infrastructure platforms, IBM's "Blue Cloud" expands the technologies of Internet to enterprise platform; Goolge's cloud platform serves for its main searching service and other applications. The cloud-processing platform in Google consists of four independent and tightly bound systems: Google File System, Distributed File System, MapReduce programming mode and distributed lock mechanism. The most important contribution in cloud programming model is MapReduce, it is a parallel programming model proposed by Google and can be applied in massive data processing. Amazon proposed Elastic Compute Cloud (EC2) and Simple Storage Service (S3) to provide enterprises with computing and storage service. In China, cloud infrastructure construction is merging, governments and enterprises have invested a lot of fund and brought out various projects such as "XiangYun" in Beijing, "YunHai" in Shanghai, "TianYun" in Guangzhou, Cloud Computing International Joint Laboratory in Shenzhen , Cloud Computing Center in Shangdong and "AliYun" of Alibaba.

3 The Architecture of Emergency Cloud Services

The emergency management architecture involves lots of profound and sophisticated scientific issues and is characterized by multi-agent, multi-inducement, multi-scale, polytrophic, multi-information[14], and so on. The cloud service architecture for unconventional emergencies aims to provide a series of emergency management service based on cloud computing technology for monitoring and early warning, information reporting, emergency responding, aftermath resolving and emergency plan revising in the lifecycle process of unconventional emergencies[8, 15]. We proposed hierarchical cloud service architecture for unconventional emergencies, including IaaS, PasS and SaaS (Figure 1) to achieve rapid deployment for resources and real-time decision support for emergency management.

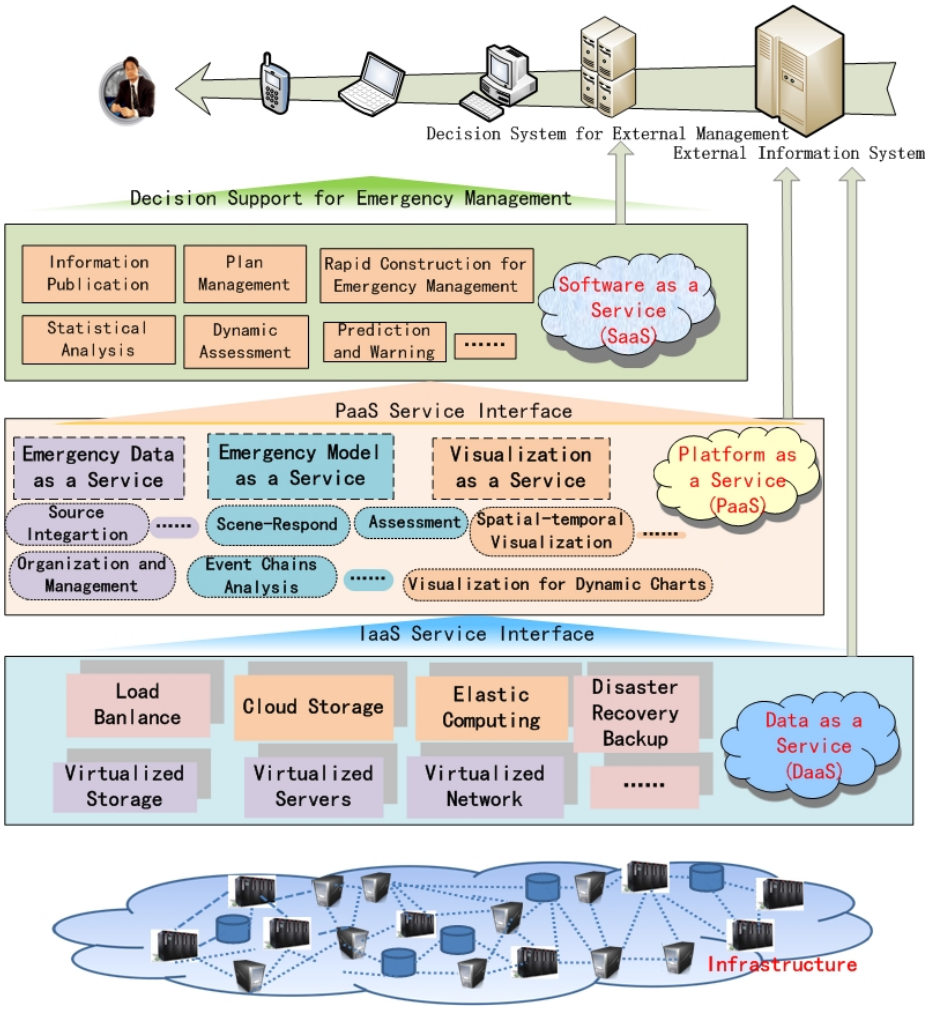


Fig. 1. Cloud Service Architecture for Unconventional Emergency Management

- Infrastructure as a Service, IaaS

Infrastructure as a service (IaaS) offers a full computer infrastructure to emergency management system, researchers and decision makers by delivering all kinds of virtualized resources via the Internet, including CPUs, storage and any other basic infrastructure. Users can access virtualized resources as a public utilize and they can deploy and run any applications without managing or controlling any infrastructures.

- Platform as a Service, PaaS

Platform as a service (PaaS) offers an open application programming interface and development platform, such as service invocation of middleware like emergency

information access, emergency model and visual representation, to rapidly develop and run their own model and emergency management system for emergency researchers.

- Software as a Service, SaaS

Software as a service (SaaS) provides up-to-date cloud-based emergency software application services over the Web for emergency management researchers, decision makers and general public. SaaS eases software maintenance and update, enable multiple clients to share a single application through a browser or a handheld device.

4 Key Techniques of the Emergency Cloud

In order to implement the above hierarchical cloud service architecture and accomplish the rapid deployment of sources and real-time decision support for emergency management, the emergency cloud service architecture includes six key techniques.

4.1 Rapid Information Retrieval from Cloud and Terminal

Unconventional emergency management often needs to gather data from different government departments in a short time because of the cross-regional characteristic of unconventional emergency. Due to distribution of the functional division of the government department, division of government network and confidentiality of information, it is difficult to access these data directly. Therefore, the study is needed of the emergency condition aggregation network of cross-departmental data from the perspective of technology and establishment of transmission channel for emergency condition data. Furthermore, with the development of the information technology, as the first witness of the emergency, general public publish the information of emergency and mass incident to mass media ,such as by means of personal micro blogs, social networks and so on, at first time with high levels of enthusiasm. The emergency information from mobile devices of general public will be an important source with the progress of the society and the participation of the public.

To resolve the problem of emergency condition aggregation of the cross-departmental data, we adopt the techniques of RDF, SPARQL, RDL and Linked Data. Specifically, we make use of RDF to format the heterogeneous emergency data and metadata； We use SPARQL, RDL to improve the query capability of each data source; we adopt the mechanism of Linked Data to construct a freely extensible and non-integrated data network for emergency management.

4.2 Infrastructure as a Service for Emergency Management

The information access of unconventional emergency is characterized by dynamic, concurrency and diversity and requires scaled I/O load. Given all the problems above, IaaS for emergency management will provide the following services for users: **Virtual machine service(VMS):** VMS provides users with a scalable, on-demanded,

safe and usage meter based computing service. Users can customize the computing resource in few minutes through the self-service system and then use the virtual machine with SSH or remote desktop connection. **Cloud storage service:** Could storage service provides users with an easy-to-use, reliable, safe, fast and extensible storage service which based on key/value pair for emergency data. Users can access the data in cloud through Internet, client or the API of the standard HTTP REST interface. **Processing service for big data:** The big data processing service provides users with a MapReduce based Web service. The provided Web service enables users to process massive data under an easy and inexpensive way.

4.3 Data as a Service

Most of the data properties of unconventional emergencies are relevant to the location in space. These data have no explicit space coordinates, but there is implicit space position information in the content of the data. The spatial and temporal attributes of emergencies are considered essential to information storage and management. The existing geographic coding system based on space places database might not performance well in geocoding for the attribute data from network[14]. Firstly, we adopt the multi-scale spatial data fusion technology to establish a scale-span spatial geographic database based on the existing basic place name database. Meanwhile we establish multiple version basic place name database and geocoding database to provide a data basis for the subsequent intelligent geocoding. In a data-intensive computing environment, we employ automatic monitoring technology for geo-spatial location to retrieve the geo-spatial key words of the text description from Web. There are inheritance and inclusive relationships between the geographic locations. We can make use of the spatial topologic relation to evaluate the uncertainty for geocoding. Secondly, we manage the emergency data in multiple dimensions, including space, time and event. The technology route adopted is using the object-oriented modeling methodology to design a time-varying model for events. We introduce the default rules of events to simplify the varying process of the events and finally improve the speed of the retrieval.

4.4 Modeling Integration Cloud Service for Emergency Management Decision Support

Emergency management models are different in the data format of input/ output, the developing language and interface. The greatest technical challenge with emergency management decision support is providing uniform componentized encapsulation and service call.

Firstly, we employ service-oriented architecture (SOA) structure model and component-based development model to accomplish the distributed deployment, and composition for the loosely coupled coarse-grained components through Internet. We combine the web applications as different service and employ web services to accomplish the component-based encapsulation for the model application tools at different level of grains with the multi-task scheduling technique. Secondly, based on the user

requirement for models, we employ dynamic replica strategy to actively copy model and make the input setting available to provide the tenants who have changes on model input parameters with privileged services. We employ a multi-channel queue manager to manage the tasks, and employ multiple data centers cooperation mechanism to deploy the application rapidly to satisfy the needs of sudden accesses. Finally, we establish a data distribution and invocation model with considering the size of the data and the computing capability of the nodes. Based on the characteristic of the model and data, we expand the MapReduce programming model to implement the extended and flexible computing for emergency application.

4.5 Visualization Cloud Service for Emergency Management Decision

In emergency systems, different analysis model aims at analyzing different specified events and also get the different analyzed result. Due to the characteristics of great capacity, complex construction of spatial-temporal data, the traditional spatial analysis methods cannot be applied well to the spatial analysis for unconventional emergencies. Firstly, we propose a cloud-based exploratory analysis framework for spatial visualization. The visualization and analysis platform is divided into three tiers, including data layer, application layer and representation layer(Figure 2). The data layer is consisted of data storage, data acquisition, data fusion and intelligent geocoding services. The application layer includes data expressing model, spatial-temporal query model, and spatial-temporal analysis computing service. Representation layer contains the user interaction for spatial-temporal analysis, parameter modulation, the input and output for model and the expansion and definition of the exploratory spatial-temporal analysis model[16]. Secondly, we provide the spatial-temporal visualization and analysis cloud service for emergency management. We fuse the heterogeneous data from different source in a same data visualization platform by considering the access, storage and fusion mechanism. We expand the web map services (WMS) based on the spatial-temporal visualization analysis platform. The WMS of open geospatial consortium (OGC) is the most widely used map service criterion. WMS provide the display of the map in ways of web services and the provided display can be easily integrated to the client applications. The current WMS protocol mainly aims at the visualization of the static and general map and it is lack of the dynamic feature and is not effective for the unconventional emergencies. In order to express the spatial-temporal features effectively, we expand the WMS's display of map and visualization analysis capability[17]. Web processing services (WPS) encapsulate the geographic computing as a standard service. The geographic computing will be called as a service. The current WPS mainly contain some classic algorithms for GIS and spatial analysis. This paper expands WPS to include not only the classic algorithms for GIS and spatial analysis but also the spatial-temporal simulate ability for dynamic events and spatial-temporal analysis ability. Thirdly, we divide the flow mapping into three relatively independent sections, including data, model and visualization. The sections connected to others in the way of WMS. The visualization section is based on HTML5 and it makes easier to cross-browser deployment and publication.

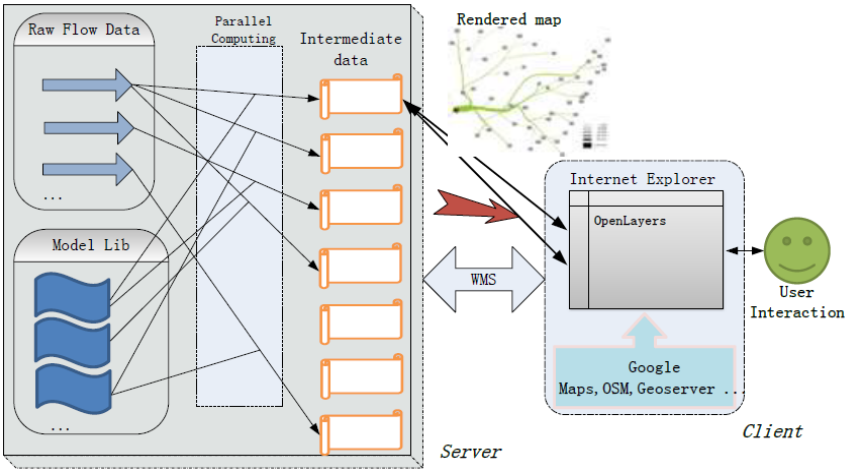


Fig. 2. Flow mapping Dynamic Visualization Analysis Framework in Big Data Environment[17]

4.6 Quick Establishment of Emergency Management System

Unconventional emergencies have the characteristics of randomness and unpredictability. Thus, new management system and decision support pattern varying with the requirements are needed. So how to call the right emergency service according to the different requirements and tasks is a key to the quick establishment of emergency management system. Given all of the above, this paper adopts the Petri net and XML to accomplish the rapid construction for emergency management, the detail implementation is as following: Firstly, we employ Petri net to represent the tasks in emergency handling process and the logic relationship between the tasks. The Petri net is a strict processing formalism and it has accurate definition and strong expression capability. It can express the control structure of sequence, parallel, circulation and condition. Thus, it can avoid the uncertainty, ambiguity and contradiction caused by other formal languages. Secondly, we employ XML to define the template of the emergency management system. XML enables the template of emergency management system to be dynamically updated. Finally, we provide text-based combination to enable users to dynamically invoke cloud service to construct emergency management system.

5 Cloud Emergency Management Scenarios

Fig 3 is the logical structure of the cloud-based service platform for emergency managing. Base on the standard of cloud emergency managing we proposed and the distributed infrastructure, cloud emergency managing will provide a lot of convenience:

- To provide IaaS service which support emergency managing through virtualization technology and cloud-base service for resource management.
- To use the cloud-base storage service to get the emergency data rapidly in both real world and the Internet.
- To re-organize all kinds of data based on their spatial and temporal property and take them into cloud emergency managing pool in PaaS form by the management of the huge amounts of data.
- To encapsulate multiple types of emergency model and take them into cloud emergency managing pool in PaaS .
- To provide space-time visualization web services as PaaS form.
- To implement emergency information dissemination and query statistics software through some technology like service search and invoke.
- To build emergency software system for specific target through the quick-build emergency system technology and provide services in the form of laaS.
- To provide as a unified access entrance for all the users through cloud emergency managing operations management, cloud-based service monitoring and self-service system.

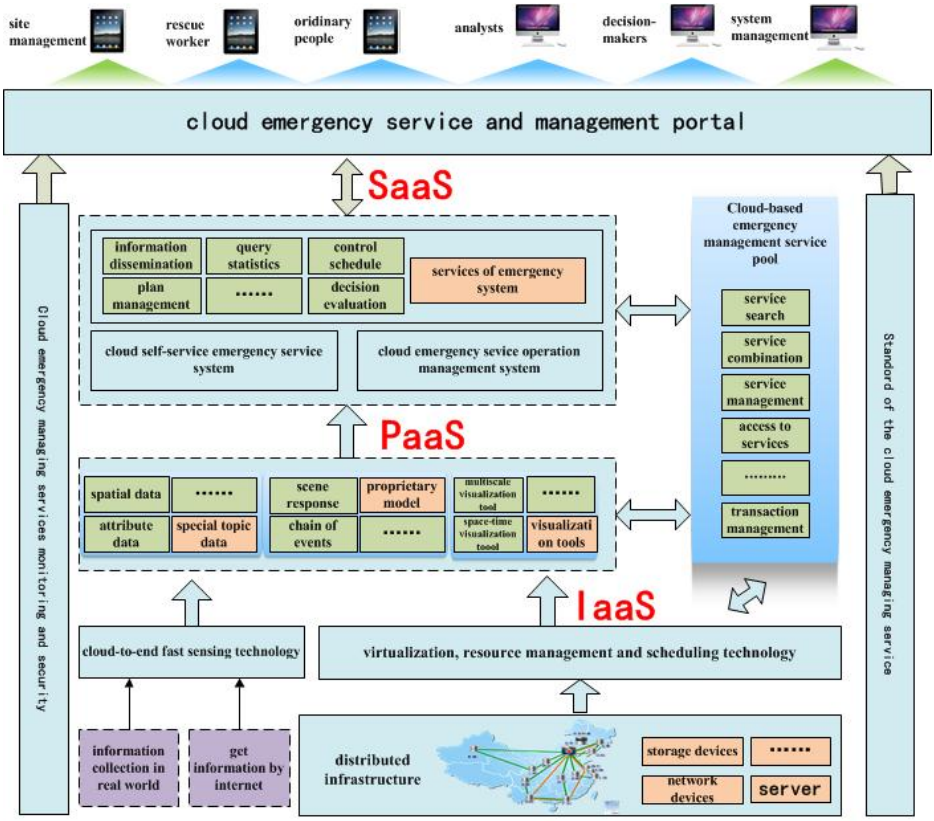


Fig. 3. Structure of the cloud-based service platform for emergency managing

We designed three scenarios to verify the effectiveness of the proposed cloud emergency managing architecture: earthquake monitoring and emergency response, economic and financial emergency managing, and emergency managing of foodborne diseases.

Earthquake monitoring and emergency response scenario take earthquake event as a sample to research on the sudden natural disaster monitoring and emergency response in cloud emergency service base on the technology: earth observation-based remote sensing data acquisition, multi-source remote sensing data fusion, massive heterogeneous spatial and temporal data analysis, real-time information push and scheduling assistive technology. This application simulates the massive remote sensing information integration, analysis and decision-making scheduling auxiliary when the earthquake suddenly attacks. This technique has been tested effective in the Wenchuan earthquake of China in 2008. It verified the quick-build capacity, storage capacity and computation scheduling of the emergency management system in cloud service system.

The scenario of Economic and financial emergency management demonstration is monitoring and pre-warning for emergencies at home and abroad of China's economic and financial system and the policy responses. The application focus on how to apply cloud services infrastructure to support the research of unconventional emergency evolution of the economic and financial system, forecast and pre-warning mode and policy simulation. On the basis of inter-ministerial boom joint meeting, the demonstration focus on the research of crisis response coordination mechanism of the National Development and Reform Commission, People's Bank of China, the Ministry of Commerce, the SAFE and other relevant departments of the country's economy. At the same time, it also provides cloud service solutions for tasks such as shared inter-department data security, complex system simulating calculation and decision-making visualization etc.

Foodborne illness emergency management is applied for Internet-scale monitoring and early warning for unconventional food safety incidents and analyzing responses from society. Based on real-time acquisition of data from open Internet, medical health, mobile terminals and food contamination and foodborne diseases, we can identify large-scale outbreak of foodborne illness, trace its root and evolution, and study the impact of emergency management due to the viewpoints of development of emergency management of the social media, and then put forward integrated mechanisms of warning and management control. So the design based on the emergency management system integrates emergency management cloud services platform for both peacetime and wartime food safety and foodborne illness and resolve continuous situation responding passively, better regulating the production management and guaranteeing people's health and social stability.

6 Conclusion

Cloud computation architecture is a new way for emergency management, and brings forward the concept of cloud emergency managing. Cloud emergency managing makes

contributions to seamless the seamless integration of heterogeneous information, the aggregation and management of emergency resources and the situational awareness, analysis and intelligent decision of the real world, as well as the cyber space. Emergency managing process is a major component of the cloud emergency managing. Cloud emergency managing will call a variety of emergency resources based on the need and help emergency department to schedule and work together in order to provide a flexible and efficient emergency service to emergency demand-side.

Acknowledgements. The work is partially supported by Natural Science Foundation of China under Grant 91224006, the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA06010202, "Twelfth Five-Year" Plan for Science & Technology Support under Grant No. 2012BAK17B01-1, the Special Project of Informatization of Chinese Academy of Sciences in the Twelfth Five-Year Plan under Grant No. XXH12504

References

1. Han, Z., W.W, Yang, L., Zhang, W.: Backgrounds,Targets, and Organization of the Major Research Plan “Study on Unconventional Emergencies Management”. Bulletin of National Natural Science Foundation of China 23 (2009)
2. Li, Q., Zhen, X.: A Survey on Cloud Computing. Computer Sciences 38, 32–37 (2011)
3. De-yi, L.: Cloud computing supports sociality, intensiveness and specialization of information service. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition) 22, 698–702 (2010)
4. Ji, X., Weng, W., Ni, S., Fan, W.: Warning classification model for public emergencies. Journal of Tsinghua University (Science and Technology) 8, 006 (2008)
5. Kaplan, R.M., Berry-Rogghe, G.: Knowledge-based acquisition of causal relationships in text. Knowledge Acquisition 3, 317–337 (1991)
6. Weichen Fan, Y.L.: Analysis of Urban Public Safety System Structure. Municipal Administration and Technology 5, 38–41 (2009)
7. Huifang, C.L.Z.Q.W.Y.X.: Research Review of Unconventional Emergency’s Theory Method Based on Context. Journal of Intelligence 30, 40–45 (2011)
8. Liu, T.: Design of the emergency plan system’s concept. Journal of Safety Science and Technology 7, 5–13 (2011)
9. Zong-chen, W.F.-y.Q.X.-g.Z.D.-j.C.Z.-d.F.: A Computational Experimental Platform for Emergency Response Based on Parallel Systems. Complex Systems and Complexity Science 7, 1–10 (2010)
10. Da-jun, D.W.C.Z.-d.Q.X.-g.W.F.-y.Z.: Semantic modeling for artificial society in parallel emergency management system. Systems Engineering —Theory & Practice 32, 1010–1017 (2012)
11. Fan, W.: Advisement and Suggestion to Scientific Problems of Emergency Management for Public Incidents. Bulletin of National Natural Science Foundation of China 2, r3 (2007)
12. Zhidong, Z.D.W.F.C.: Open Source Information in Emergency Response. Science & Technology Review 26, 27–33 (2008)
13. Fan, W., Yuan, H.: Status analysis and countermeasures for emergencies platform construction. Informatization Construction 9, 14–17 (2006)

14. Soderland, S.: Learning information extraction rules for semi-structured and free text. Machine Learning 34, 233–272 (1999)
15. Wang, H., Cao, C., Gao, Y.: Design and Implementation of a System for Ontology-Mediated Knowledge Acquisition from Semi-Structured Text. Chinese Journal of Computers 28, 2010–2018 (2005)
16. Anselin, L., Sridharan, S., Gholston, S.: Using Exploratory Spatial Data Analysis to Leverage Social Indicator Databases: The Discovery of Interesting Patterns. Social Indicators Research 82, 287–309 (2007)
17. Guo, D., Wu, K., Zhang, Z., Xiang, W.: WMS-based Flow Mapping Services. In: IEEE Services 2012. IEEE, Hawaii (2012)

Exploiting Content-Based Pub/Sub Systems for Event Detection in Emergency Management

Yuwei Yang, Beihong Jin, Fusang Zhang, and Sen Li

Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
University of Chinese Academy of Sciences, Beijing 100190, China

Abstract. Emergency management needs to detect diverse events, covering non-spontaneous events and spatio-temporal events, and notify the related personnel as soon as possible so that they can make further decisions. This paper introduces how to carry out event detection and notification by exploiting content-based Pub/Sub systems, and it also summarizes our work progress in this field. Through describing our event detection language and its application examples, the paper demonstrates the event detection capability of our Pub/Sub systems. Furthermore, this paper explains subscription/event routing strategies employed in our Pub/Sub systems constructed in different network environments (including cloud and VANETs), and these strategies serve for dynamic load balance and efficient event notification, respectively.

1 Introduction

Emergencies refer to the events which have extensive negative effects on society and pose serious threats to the life and property. Emergencies can be divided into four categories, that is, natural disasters, accidents, public health events and social security events. Emergency management aiming at large scale emergencies consists of a series of activities such as emergency prevention, detection, forecast, preparation, handling, aid and so on.

The biggest challenge of emergency detection lies in the uncertainty of emergencies. When some emergency happens, its cause and law of development are unknown. Therefore, no useful experience in dealing with emergencies leads to casualties, property damages and other catastrophic effects.

Although it is impossible to predict the occurrence of emergencies, we note that a balanced status in the environment must be broken once some emergency happens. As for now, various devices including different kinds of sensors and RFID devices make it possible to be aware of the environmental status, and diverse connected networks make it convenient and effective to transmit the environmental status. It is expected to detect emergencies by monitoring the changes of the environmental status. In particular, if some regular event does not happen, it can be deemed to be an irregular event (i.e., an exceptional event). To be more general, emergencies can be defined as the events that have deviations with regular events. However, deviations are difficult to be depicted in advance and they are always related to applications. For instance, if a trajectory has the same initial point as some other trajectories but does not coincide with most of them, it

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 199–210, 2013.

might be treated as exceptional, but the deviation degree between trajectories used for judging exceptions depends on specific applications. Obviously, in scenarios of monitoring cash trucks and distribution of emergency goods, the deviation tolerances of vehicle trajectories are different.

Pus/Sub systems can be used to identify and distribute varieties of events for interested users. In a content-based Pub/Sub system, events reflect object status and they are represented using conjunctions of pairs of attributes and their corresponding values. The above mentioned regular events that do not happen are named non-spontaneous events, and they cannot notify their occurrence by themselves. For example, if some cash truck does not go along the assigned route in the specified time interval, this event is regarded as a non-spontaneous event. When this kind of event happens, a Pub/Sub system will notify the related personnel.

Furthermore, after emergencies occur, it is especially important to ensure the goods used for escaping, saving, aiding, and keeping alive. Emergency management requires an effective emergency logistics system. In emergency logistics, it is often needed to detect the events which have spatio-temporal constraints, that is, spatio-temporal events. Some examples include that whether a specific vehicle leaves an assigned warehouse, whether a specific vehicle arrives in an appointed area in a given time period, whether a fleet arrives at multiple unloading places in a scheduled order, which products need to be replenished from a certain warehouse and unloaded at the site allocated for relief supply distribution, etc. Gaining the information about goods arrival, storage and on passage in time contributes to identifying potential risks as early as possible and optimizing goods scheduling.

Pub/Sub systems can be employed to monitor the current status of goods and notify the related personnel of the corresponding events so as to provide a basis for effective goods scheduling. In general, a Pub/Sub system serving for emergency logistics can be constructed on the cloud infrastructure located at various levels of emergency centers. This kind of design is beneficial in handling potentially wide variation in workloads in emergency scenarios. In addition, it is noted that in emergency scenarios, vehicles equipped with onboard-units can constitute a Vehicular Ad hoc NETwork (VANET), which makes it possible to transmit data over long distances with the help of RSUs (RodeSide Units). A Pub/Sub system over a VANET can monitor current status of goods and notify vehicles of real-time road conditions and/or traffic control messages. If some road segments are forbidden to go through due to a sudden accident, vehicles and RSUs around the accident location can publish the corresponding events so that the vehicles receiving corresponding notifications can bypass those roads in advance.

So far, we have developed Grus [1], a content-based Pub/Sub system for Internet of Things, OPS4Cloud [2] and APUS [3], two content-based Pub/Sub systems on the cloud infrastructure and VANETs, respectively. These Pub/Sub systems can be used to serve for different emergency scenarios.

The rest of this paper is organized as follows. Section 2 overviews the work related with Pub/Sub systems and exceptional event detection. Taking our developed Pub/Sub systems as examples, Section 3 demonstrates the capability of detecting complex events, and Section 4 presents subscription/event routing strategies in Pub/Sub systems constructed in different network environments (including cloud and VANET). The last section makes conclusions of this paper.

2 Related Work

In a content-based Pub/Sub system, a composite subscription language, also called as an event detection language, is used for describing the conditions that events to be detected should satisfy, namely, the patterns to filter data. The roadmap of event detection languages can be characterized by the capability of depicting constrains between events. An event detection language can only express logical constraints between events at first, and temporal/spatial constraints later on. Afterwards, the expressiveness of spatio-temporal constraints is provided. For instance, Padres [4] is a Pub/Sub system for workflow management and business process execution, and provides a composite subscription language to support composite subscriptions in four kinds of event relationships, that is, "and", "or", "happen sequentially", and "happen repeatedly". [5] starts from the requirements of RFID applications, concerns the temporal relationships between events, especially the temporal relationships in strict partial orders, and supports several types of events such as logical events, temporal events (including non-spontaneous events) and iteration events. [6] provides the identification and processing of spatial events for a smart location service by designing and implementing two predicates, i.e., "Within" and "Distance". [7] discusses the requirement for a spatial alarm, that is, a certain user is allowed to specify his/her interested spatial region in advance, and the alarm will be triggered once some moving object enters the specified region. In essence, this requirement is to detect spatial events. [7] and [8] present the solutions to spatial alarms, and introduce the safe interval and the safe region to accelerate the processing.

Pub/Sub systems on the cloud infrastructure are implemented through transplanting existing systems (e.g., [9]) or developing new ones, such as Amazon SNS (Simple Notification Service) [10] and YMB (Yahoo Message Broker). The latter can guarantee that published messages will be delivered to all the subscribers, even in the presence of certain broker failure. It is used as the message broker component in PNUTS [11], but it is a topic-based Pub/Sub system.

As a pioneering Pub/Sub system over VANETs, [12] proposes to deliver an event to multiple homeZones in a designated area and store them for a certain time period, and there RSUs in the area (if present) or vehicles travelling within the area are selected as homeZones to keep the events. As thus, when homeZones receive the subscriptions broadcasted by the vehicles travelling nearby, they perform the matching with the stored events and notify the vehicle users of the events if matched. However, [12] imposes the restrictions on valid areas of notifications. In other words, the vehicles out of this area cannot receive any corresponding notification even if they show interest in this event. In [13], not only the valid areas of notifications but also the event types are restricted. In detail, [13] advertises an event type and a notification area in advance, and then marks the subscriptions within the notification area valid. If some vehicle drives off the area, the subscriptions carried by the vehicle will be regarded as invalid. As a result, only the vehicles whose users subscribe this kind of events and move within this area can receive the corresponding notifications.

It is important to note that there are varieties of techniques to detect unknown events. If we assume that the occurrence frequency of normal events is much higher than that of exceptional events, clustering algorithms can be utilized to judge the events which

are far away from the clustering centers as exceptional events. Moreover, we can obtain event patterns through learning, and identify exceptional events based on these patterns. In this process, the machine learning algorithms, data mining techniques, statistics and information theory are often required.

The challenges of detecting exceptional events out of the continuous data flows come from two aspects: on one hand, the continuously increasing data makes it hard to save historical data completely; on the other hand, the knowledge contained in the data will evolve constantly with the arrival of new data. Some researches have targeted at these issues. For example, [14] proposes to assign a weight to a classifier based on its expected classification accuracy on the test data and then use weighed averaging of the outputs of the classifiers so as to improve the prediction accuracy. However, generally speaking, there is still a long way to go.

3 Event Detection Language

We have developed Grus, a content-based Pub/Sub system for Internet of Things. In Grus, we design an expressive event detection language EDL to specify spatio-temporal events. For a system of detecting spatio-temporal events, the way of specifying time and locations of events is its foundation. We adopt an interval to denote the duration of event occurrence, using the event attributes *stm* and *etm* to denote an event's start and end time, respectively. Meanwhile, we treat the geographical location where an event occurs as a simple region (i.e., a region without holes), and use as an event spatial attribute a set of points which describe the region's boundary in the geographical map and the minimum convex polygon which contains all points in the set.

3.1 Operators and Subscription Variables in Composite Subscriptions

The event detection language EDL adopts the content-based subscription scheme. In brief, a primitive subscription in EDL is still a conjunction of predicates, but predicates are in the form of "type: name operator value". A composite subscription in EDL is still composed of several constitutive subscriptions, but the operators which connect constitutive subscriptions are newly-defined ones. EDL provides the following temporal operators:

- negation operator (in the basic form of $!(T;t)$, or its variants $!(A;t)$ and $!(A;B)$)
- happen-between operator (in the basic form of $=|(T;t)$, or its variants $=|(A;t)$ and $=|(A;B)$)
- happen-after operator (in the basic form of $|=(T;t)$, or its variants $|=(A;t)$ and $|=(A;B)$)
- concurrent operator (in the form of $|(A)$)

where T stands for an actual time point, and t denotes a temporal interval expressed by an integer, and A and B denote subscriptions, which, in the above operators, convey the time points of the occurrences of events satisfying A or B.

The choice of the above temporal operators is the result of synthesizing theory results with practical needs. The classical interval algebra [15] has recognized a total

of 7 temporal relations between the temporal intervals in which the events occur, i.e., precede, meet, overlap, cover, start, finish, and equal. From a practical point of view, we refine "precede" relation by distinguishing "happen-between" from "happen-after" relation, and merge "overlap", "cover" and the others into "concurrent" relation. Next, the negation operator is introduced for observing non-spontaneous events. Finally, by providing the basic forms of the above temporal operators, the subscriptions in EDL are permitted to bind to specific time points.

The choice of spatial operators follows a similar working style. Spatial relationships between two regions can be classified into topological relationships, direction relationships, and metric relationships [16]. First, the topological relationships over regions can be obtained by enumerating the intersections of boundaries and interiors of two regions. The most famous theory, by far, is the spatial logic RCC-8 [17], which provides eight kinds of spatial topological relationships, that is, if there are two regions called R_A and R_B, then one of the following relations holds: R_A disjoints R_B, R_A externally connects R_B, R_A overlaps R_B, R_A equals R_B, R_B internally contacts R_A, R_B in R_A, R_A internally contacts R_B, and R_A in R_B. In accordance with RCC-8 and from a practical point of view, we provide three kinds of spatial operators, i.e. "overlap", "happen-in", and "same-place", where "overlap" has the same meaning as "overlap" plus "externally connect" in RCC-8, "happen-in" has the same meaning as "in" plus "internally contact" in RCC-8, and "same-place" has the same meaning as "equal" in RCC-8. Next, spatial metric relationships include the area of a region, the perimeter of a region, etc. We provide the operator for expressing the distance between the locations of two events, i.e., Euclidean distance between the centers of two areas where events take place. Now the distance operator is designed to work only on the condition that the locations of the two events are disjoint, so we can omit the disjoint operator. Finally, for the time being, we ignore the spatial direction relationships. From our observation, although the spatial direction relationships can help declare the spatial relationship between two disjoint areas, the usages of such relationships are limited in IoT applications. As a result, EDL provides the following spatial operators:

- happen-in operator (in the basic form of *@(R)*, or its variant *IN (A)*)
- overlap operator (in the basic form of *OVLP(R)*, or its variant *OVLP (A)*)
- same-place operator (in the basic form of *SPL(R)*, or its variant *SPL (A)*)
- distance operator (in the basic form of *DIST(R;<;l)*, *DIST(R;=;l)*, or *DIST(R;>;l)* or their variants *DIST(A;<;l)*, *DIST(A;=;l)*, or *DIST(A;>;l)*)

where *R* denotes the region represented as a convex polygon, *l* is an integer denoting the distance and *A* denotes a subscription, which, in the above operators, conveys the location of the occurrence of event satisfying *A*.

EDL also defines two logical operators *&&* and ||, which denote logical "AND" and logical "OR" between two events, respectively. Finally, same as in [5], EDL defines operators P(t), Q(n), and S(n, t, k) so as to specify different iteration events.

In order to express the correlations among different subscriptions in one composite subscription, in particular, to express the concurrent spatial and temporal relationships among different events, we introduce the concept of subscription variable. A subscription variable is the symbol associated with a subscription. In a composite subscription, a subscription variable begins with "*$*", ends with "*;*", and can show up in the same position where a subscription may appear. Assuming that, in composite

subscription *A*, there are two subscription variables "*$var1;*"and "*$ var2;*" we have to conform to the following form: *A, $var1;=(sub1), $var2;=(sub2)*, where "*$var1;*"and "*$var2;*" are bound to subscriptions *sub1* and *sub2* respectively, and they can occur in such a place within *A* where a subscription may appear.

3.2 Examples

We note that the motivation behind observing spatio-temporal events comes from the fact that the objects are in constant motion. The objects may move around as time elapses, but all the changes of topological relations between two objects are reduced to a total of six modes [18]: LEAVE, HIT, REACH, EXTERNAL, INTERNAL, and CROSS. We can decompose such a motion mode into several related events happening within some interval and then capture them by submitting subscriptions in EDL. In the following, we illustrate the usage of EDL through a specific application scenario in which the motions of cash trucks are detected.

As we know, managers in banks need to monitor whether trucks loaded with cashes leave the bank coffers, and whether they reach the destination subbranches. Suppose that (1) the bank coffer locates in *Region1* and the destination subbranch locates in *Region2*, (2) subscription *A* concerns the events of occurrences of a certain truck. As thus, Subscription1 in the below can be used to observe the events that trucks locate originally in *Region1*, and after a period of *t1* they occur at a place to which the distance from *Region1* is more than *x1*. If some events satisfying Subscription1 are detected, it means that the trucks have left the bank coffer. Regarding the bank coffer as a static object, Subscription1 describes the LEAVE moving mode of a moving object and a static one. On the other hand, Subscription2 is used to observe a kind of non-spontaneous event that trucks locate originally in a place more than *x2* far from *Region2*, and do not arrive at *Region2* during the period of *t2*, that is, Subscription2 can examine the trucks which do not reach the destination subbranch.

=|(@(Region1)(A);t1)(DIST(Region1;>;x1)(A)) (Subscription1)
!(DIST(Region2;>;x2)(A) ; t2)(@(Region2)(A)) (Subscription2)

In the cash truck transportation process, the distance between the head cash truck and another cash truck *CT* should never be more than *x*. If the distance is beyond *x*, then it means *CT* is deviating from the fleet of the trucks, and the related managers are supposed to receive the notification for the emergency. Therefore, the managers can submit subscriptions in the following format to observe the LEAVE mode of two moving objects:

=|($e1;; t)($e2;),
$e1;=(|($A;)($B;) && DIST($A;;<;x)($B;)),
$e2;=(|($C;)($D;) && DIST($C;;>;x)($D;)),
$A;=E-A, $B;=E-B, $C;=E-A, $D;=E-B (Subscription3)

where subscription *E-A* concerns the events of occurrences of the other cash trucks except the head cash truck, and subscription *E-B* concerns the events of occurrences of the head one. Subscription3 is used to observe the events that the distance between the head cash truck and a certain other one changes from less than *x* to more than *x* in the period of *t*, which means the corresponding cash truck is deviating from the fleet

of the trucks. Once the above events are detected, the corresponding event notifications will be delivered to the related managers.

As another example in the transportation process, the distance between any two cash trucks, named *CT1* and *CT2*, should never be less than *y* which is supposed to be the safe distance. If the distance is less than *y*, then it means that an accident might happen between *CT1* and *CT2*, and the drivers of the two trucks are supposed to receive the notification for the emergency. Therefore, the drivers can submit subscriptions in the following format to observe the HIT mode of two moving objects:

=|($e1;;t)($e2;),
$e1;=(|($A;)($B;) &&DIST($A;;>;y)($B;)),
$e2;=(|($C;)($D;) && DIST($C;;<;y)($D;)),
$A;=E-A, $B;=E-B, $C;=E-A, $D;=E-B (Subscription4)

where subscription *E-A* concerns the events of occurrences of the other cash trucks except the local cash truck (here, we treat the cash truck driven by the subscriber as local cash truck), and subscription *E-B* concerns the events of occurrences of the local one. Subscription4 is used to observe the events that the distance between the local cash truck and a certain other one changes from more than *y* to less than *y* in the period of *t*, which means the distance between these two trucks is under the safe threshold. Once the above events are detected, the corresponding event notifications will be delivered to the related drivers.

4 Routing Strategies

The subscription/event routing strategy is the key to detecting distributed events efficiently. In this section, we present the different routing strategies in OPS4Cloud and APUS, respectively.

4.1 Routing Strategies in OPS4Cloud

System Overview
OPS4Cloud adopts the client/multi-server architecture. Servers, also called brokers, are in charge of subscription management, event detection and notification. In OPS4Cloud, brokers are grouped into multiple regions. In each region, one broker is designated as *master* and the others as *salves*. Here, the master, which fully connects with the masters in the other regions, is responsible for maintaining the routing table and routing the advertisement, subscription and event messages, and the salves, linking with their corresponding master, keep the same subscriptions as the master. However, deciding which broker (master itself or one of the slaves) is responsible for matching specific events with some subscriptions is still the duty of the master.

Subscription Organization and Advertisement Support
In OPS4Cloud, subscriptions on a broker are organized into subscription trees where a node denotes a subscription and an edge denotes the covering relationship of subscriptions in two connected nodes. We use the vector of predicates of a subscription as the event space of the subscription, and define the event space of a subscription tree

as the event space of a root subscription. Thus, we can get the distance between any two event spaces of subscription trees by calculating the distance between the corresponding predicate vectors. For multiple trees on a broker, if the distance between any two event spaces is close enough, we can get a Composite Event Space (CES) of these trees by merging the same component of the predicate vectors of two roots in turn. In general, there are multiple CESs on a broker. Specially, a union set of all CESs on a broker is called the composite event space of the broker (CES-b), and a collection of several CESs is called a CES set (CES-Set). Any CES is marked with a unique identifier, i.e. Composite Event Space ID (CES-ID).

In OPS4Cloud, advertisements are allowed to be propagated before events are published by publishers. Advertisements, with the similar specifications and event space expressions to subscriptions, describe the publishers' intentions to publish a particular kind of events, so each advertisement is required to contain a predicate indicating the type of events to be published. In addition, each advertisement is assigned with a unique advertisement ID, and thus an event can designate an advertisement ID to show the advertisement which it belongs to. One or more masters are designated for holding an advertisement, and then the covered subscriptions are aggregated to those masters. This reduces forwarding times of events in the matching process, resulting in the decrease of event response time and the increase of system throughput.

Advertisement Routing

When a master receives an advertisement message, it will select an appropriate master (supposing it is *M-A*) to hold this advertisement according to the load condition, and then forward the message to *M-A*. *M-A* will update the routing table according to the event type of this advertisement, including the following two cases:

Case 1: If the event type has existed in the routing table, a new advertisement record is inserted under the corresponding event type, and then if the new advertisement covers the root of an existing subscription tree, the event space of the tree is added into the corresponding CES-Set item of the advertisement. Finally, if the event space of the covered root originally is subordinate to the default advertisement, the event space is removed from the corresponding CES-Set item.

Case 2: If the routing table has not contained the event type, a new event type record is inserted, and two advertisement records under the event type, i.e. the received advertisement record and the default one, are constructed. The event spaces implied by the two records complement with each other and constitute a global event space under the event type.

Afterwards, *M-A* will send the updated part of the routing table to the other masters which will update their routing tables accordingly.

Subscription Routing

When a master receives a subscription, it will execute the following steps:

Step1: If the event type contained in the subscription cannot be found in the routing table, this subscription will be ignored directly as publishers are not supposed to send this type of events, otherwise, certain advertisements must overlap with the

subscription. Therefore, from those brokers whose CES-bs are subordinate to the event spaces of the advertisements, a broker denoted as *M-S1*, is randomly selected as the destination of the subscription.

Step2: *M-S1* calculates the distance from each CES under the corresponding advertisement to the received subscription, and obtains the minimum distance *minD* and the corresponding CES *mCES*. The holding broker of *mCES*, named *M-S2*, is responsible for storing this subscription. If the subscription leads to the change of the CES-Set holding *mCES*, *M-S1* informs all the brokers whose CES-bs are subordinate to the event space of this advertisement of updating their CES-Set items, otherwise, only delivers the subscription to *M-S2*.

Step3: When *M-S2* receives the subscription, *minD*, and *mCES*, it will insert the subscription into the local subscription forest in accordance with the received information and update event space records if necessary.

Event Routing

When a master receives an event message, it searches the advertisement records according to the specified advertisement ID or event type. If the advertisements covering the event are found, the event is routed to the brokers which are responsible for the advertisements. Those brokers will find out which event spaces the event belongs to, and once again route the event to the brokers which the event spaces belong to. The event will finally be routed to some masters, and the masters receiving the event will select themselves or a certain slave in their regions to execute the event matching according to the load conditions.

4.2 Routing Strategies in APUS

System Overview

We assume that RSUs are deployed on the main intersections and they hold the initial layout of RSUs. In addition, they are equipped with GPS receivers and digital maps. So, they can know from the map an optimal path from a source RSU to a destination RSU, along with locations of RSUs. In APUS, matching subscriptions and events is performed on RSUs, as well as calculating S-RSUs and N-RSUs for a subscription.

In regards to the events and subscriptions in APUS, two types, namely, LOCATION and TIME, are introduced as the types of event attributes and subscription predicates. The LOCATION type has two subtypes: REGION and TRACE. The domain of REGION contains all the convex polygons represented by a set of vertices, while the domain of TRACE includes the trajectories denoted by a list of points. In particular, in a subscription, the predefined predicate names “occ-location” and “rep-location” of LOCATION type specify an event occurrence location and an event reporting location, respectively. Similarly, the predefined predicate names “occ-time” and “rep-time” of TIME type are used to specify the constraints of event occurrence time interval and event reporting time point, respectively. Besides, a notification area and a valid notification interval can be designated in a subscription.

Subscription Routing

The vehicle generating a subscription will hand it over to the firstly-encountered RSU. Next, the RSU parses the predicates of LOCATION type in the subscription and gets a region or a trajectory of interested events, thus it obtains a list of RSUs referred to as S-RSUs located inside/near the region or the trajectory. Assuming that there are multiple RSUs, one is chosen as a master RSU randomly, while the others are regarded as mirror RSUs. Then, the RSU extracts the notification area from the received subscription and computes a list of RSUs referred to as N-RSUs within/near the notification area through the digital map. The calculation of N-RSUs is similar to that of S-RSUs. Finally, the subscription message containing the subscription and the lists of N-RSUs and S-RSUs will be sent to the master RSU by underlying data delivery mechanism.

Upon receiving the subscription message, the master RSU computes whether its neighbor RSUs are in the list of S-RSUs. If so, the master RSU forms a diffusion message for this subscription and spreads the diffusion message to the neighbor RSUs. If the mirror RSU receives the diffusion message for the first time, it will propagate the message again, otherwise, it will discard the message.

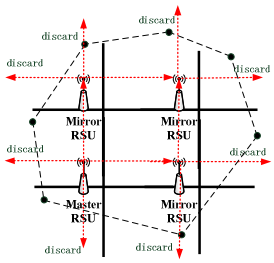


Fig. 1. A subscription propagated in a region

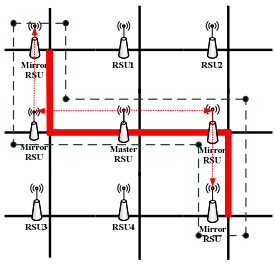


Fig. 2. A subscription propagated along a trajectory

As shown in Fig.1, a dotted line denotes the region which contains four RSUs: one is master RSU and the other three are mirror RSUs. When the master RSU receives subscriptions, it will propagate to the other three RSUs. In Fig. 2, a red bold line denotes a trajectory concerned by a subscription. Along the trajectory, there are five RSUs, of which one is the master RSU and the other four are mirror RSUs. The subscription is propagated from the master RSU to the mirror RSUs along the trajectory.

In the end, both master RSU and mirror RSUs store the subscriptions into their local matching structures and wait for the events to be matched.

Calculating S-RSUs for a Subscription

If the events which a subscription concerns occur in the region X whose boundary is convex polygon Y, then the following steps are performed to get a list of S-RSUs:

Step1: Find the farthest two points on Y and calculate the distance between them (denoted by d), as shown in Fig. 3(a).

Step2: For each edge in Y, construct a rectangle, taking an edge of Y as a side and d as the length of the adjacent side (See Fig. 3(b)), and then extend the side opposite to the

edge of Y in the rectangle. As shown in Fig. 3(c), a list of junctions can form a new convex polygon Y_e. S-RSUs are the RSUs located in Y_e.

Step3: If there is no RSU in Y_e, Y_e is expanded again and again through Steps 1-2 until there is at least one RSU in the expanded region.

Step4: If there is at least one RSU in Y_e, Y_e has to be expanded through Steps 1-2.

If locations of the events which a subscription concerns are along a trajectory, then S-RSUs are the RSUs along the trajectory. If there is no RSU along the trajectory, which means that the trajectory is only a part of a road segment, then the trajectory is extended to intersections along the road segment.

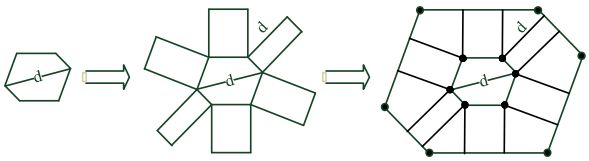


Fig. 3. (a) Polygon Y **(b)** Y with extended rectangles **(c)** new polygon Y_e

Event and Notification Routing
Routing an event is quite straightforward. The event published by a vehicle is delivered to the first-met RSU and then forwarded to the RSU nearest to the place where the event occurs through the underlying delivery mechanism, and the event published by an RSU will stay there. When an arriving event matches a stored subscription on an RSU, an event notification is sent to N-RSUs of the subscription. The whole procedure is the same as the subscription routing. When one of N-RSUs receives a hello message from the subscriber vehicle, it sends all the corresponding notifications to the vehicle and then notifies the other N-RSUs to clear the corresponding notifications. If N-RSUs find that the notifications are beyond the valid notification interval, they will delete these notifications. If a subscriber vehicle meets the vehicle carrying the notification before it arrives at the notification area, then the subscriber vehicle will receive the notification and the notification will not be forwarded further.

5 Conclusion

In the paper, we reveal the different roles played by content-based Pub/Sub systems in the scenarios of emergency management. At least, the Pub/Sub systems which have the ability to detect spatio-temporal events can help discover the emergencies and monitor the logistic status. Considering that the timeliness is critical to emergency management, our next-phase work will aim to enhance the timeliness of event detection and notification.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grants No. 91124001.

References

1. Jin, B., Zhuo, W., Hu, J., Chen, H., Yang, Y.: Specifying and Detecting Spatio-Temporal Events in the Internet of Things. Decision Support Systems (2013), http://dx.doi.org/10.1016/j.dss.2013.01.027
2. Fang, W., Jin, B., Zhang, B., Yang, Y., Qin, Z.: Design and Evaluation of a Pub/Sub Service in the Cloud. In: 2011 International Conference on Cloud and Service Computing (2011)
3. Zhang, F., Jin, B., Zhuo, W., Wang, Z., Zhang, L.: A Content-Based Publish/Subscribe System for Efficient Event Notification over Vehicular Ad hoc Networks. In: The 9th IEEE International Conference on Ubiquitous Intelligence and Computing (2012)
4. Li, G., Jacobsen, H.-A.: Composite Subscriptions in Content-Based Publish/Subscribe Systems. In: Alonso, G. (ed.) Middleware 2005. LNCS, vol. 3790, pp. 249–269. Springer, Heidelberg (2005)
5. Jin, B., Zhao, X., Long, Z., Qi, F., Yu, S.: Effective and Efficient Event Dissemination for RFID Applications. The Computer Journal 52(8) (2009)
6. Chen, X., Chen, Y., Rao, F.: An Efficient Spatial Publish/Subscribe System for Intelligent Location-Based Services. In: ACM DEBS (2003)
7. Bamba, B., Liu, L., Yu, P.S., Zhang, G., Doo, M.: Scalable Processing of Spatial Alarms. In: Sadayappan, P., Parashar, M., Badrinath, R., Prasanna, V.K. (eds.) HiPC 2008. LNCS, vol. 5374, pp. 232–244. Springer, Heidelberg (2008)
8. Bamba, B., Liu, L., Iyengar, A., Yu, P.S.: Distributed Processing of Spatial Alarms: A Safe Region-based Approach. In: International Conference on Distributed Computing Systems (2009)
9. Zhang, B., Jin, B., Chen, H., Qin, Z.: Empirical Evaluation of Content-based Pub/Sub Systems over Cloud Infrastructure. In: The 8th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (2010)
10. Amazon SNS, http://aws.amazon.com/sns/
11. Cooper, B.F., Ramakrishnan, R., Srivastava, U., Silberstein, A., Bohannon, P., Jacobsen, H., Puz, N., Weaver, D., Yerneni, R.: PNUTS: Yahoo!'s Hosted Data Serving Platform. Proceedings of the VLDB Endowment, 1(2) (2008)
12. Leontiadis, I., Costa, P., Mascoloc, C.: A Hybrid Approach for Content-Based Publish/Subscribe in Vehicular Networks. Pervasive and Mobile Computing 5(6) (2009)
13. Wu, L., Liu, M., Wang, X., Chen, G., Gong, H.: Mobile Distribution-Aware Data Dissemination for Vehicular Ad Hoc Networks. Journal of Software 22(7) (2011)
14. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining Concept-Drifting Data Streams Using Ensemble Classifiers. In: ACM SIGKDD 2003 (2003)
15. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26(11), 832–843 (1983)
16. Guting, R.H.: An Introduction to Spatial Database Systems. The VLDB Journal 3(4), 357–399 (1994)
17. Randell, D.A., Cui, Z., Cohn, A.G.: A Spatial Logic based on Regions and Connection. In: The 3rd International Conference on Knowledge Representation and Reasoning, pp. 165–176 (1992)
18. Muller, P.: Topological Spatio–Temporal Reasoning and Representation. Computational Intelligence 18(3), 420–450 (2002)

Webpage Mining for Inflation Emergency Early Warning

Yan Qu, Wei Shang, and Shouyang Wang

Academy of Mathematics and Systems Science, Chinese Academy of Science
No. 55 Zhongguancun East Road, Haidian District, Beijing, China, 100190
quyan10@mails.ucas.ac.cn, {shangwei,sywang}@amss.com

Abstract. Serious inflation turbulence is a signal of potential financial crisis and social economic emergencies. Macroeconomic early warning of inflation and other major economic indicators is critical to discover potential crisis in advance. Traditional early warning methods are based on official economic statistics which are usually either released at least one month later than the economic activities actually occur or lack of flexibility to cope with fast business changes. With proper extraction and aggregation, huge amount of Internet data can serve as a new complementarity source to facilitate more timely and accurate emergency early warning. This research innovatively adopts web data processing techniques and text mining methods to extract useful information from huge amount of Internet news reports. Based on the extracted information, a price sentiment index is proposed to detect turning points of inflation efficiently. Empirical evaluation proved that the price sentiment index is efficient in inflation emergency early warning.

Keywords: sentiment analysis, web mining, macroeconomic early warning, Consumer Price Index.

1 Introduction

Inflation is an important macroeconomic indicator reflecting the price of products and labors. Serious inflation turbulence could be a signal of potential financial crisis and social economic emergencies. Thats why inflation emergency management becomes one of the most important macroeconomic aspects in most countries. Especially in China, high inflation rate accompanied with economic growth become a major macro-control target of Chinese government. During the 2007-2009 sub-prime crisis, inflation rate in China experienced the most serious fluctuations and have reached the highest point of 8.7% and the lowest point of -1.8% since the year 2000. Inflation turning points detection can provide valuable information to crisis response and economic emergency management.

Inflation emergency early warning contains the processes of forecasting consumer price development and changes, detecting price imbalances and alerting in advance. In inflation emergency early warning, time plays a decisive role. With effective early warning, government agencies can design and implement adjusting

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 211–222, 2013.

measures to avoid the adverse consequences of economic imbalances and strong economic fluctuations. Lack of timely early warning may cause irreparable damage to the whole economy.

Traditional early warning methods are mostly based on officially published statistical indicators, which have several disadvantages. Firstly, they have a slow publication scheme, which means they are always published months later than the time when economic activities actually happen. Secondly, these data do not reflect well the structural changes in the economy. Structural changes mean the revolutionary changes in the economy that are hardly represented by those established statistical indices. And thirdly, there are still many economic aspects that cannot be covered by these indicators.

With the explosion of information, the Internet has become one of the most important accesses to information. We can obtain large amounts of information from the Internet without region, time or format limitation. Because of its timeliness, easy acquisition and wide coverage, Internet information provides a possible solution to facilitate more timely and comprehensive macroeconomic early warning, and economic indicators extracted from the Internet can serve as complementarities to regularly released official statistics.

This research innovatively adopts web data processing techniques and text mining methods to extract useful information from huge amount of Internet news reports. In order to detect turning points of inflation rate efficiently, this paper proposed a research framework to construct a price sentiment index from Google news search engine. On the basis of the proposed framework, an empirical evaluation is designed to test the validity of proposed price sentiment index in inflation emergency early warning. Conclusions and discussions are given at the end of the paper.

2 Literature Review

The increasingly widespread Internet information provides rich sources for academic studies. As a main category of Internet information, news reports have both informational and affective features (A Devitt, K Ahmad, 2007). Information feature means that news reports reflect real economic situation, and affective feature refers to news affecting the markets significantly, like stock price, trading volumes, or firm earnings. Studying these affective contents of text is referred to as sentiment analysis.

Sentiment analysis studies can be classified into three categories. The first category mainly focuses on sentiment or subjectivity classification in text (E Riloff, J Wiebe, 2003), which is identifying a text to be objective or subjective. The second category distinguishes news sentiment as positive or negative (Y, Y Zhang, H Chen, 2010). The third category, detecting strength of the opinion, enables extracting more abundant information from text (D Kaufer, C Mellon, 2000).

Sentiment analysis has been used in large-scale domains including market reacts to news reports, products reviews classification and stock price prediction.

A Devitt (2007) proposed a cohesion-based algorithm for sentiment polarity detection in financial news. PD Turney (2002) presented a simple unsupervised learning algorithm to classify products reviews as recommended or not recommended. Davis et al. (2006) examined the effects of optimistic and pessimistic language used in earning press releases on future firm performance. R Sanjiv(2004) combined different classifier algorithms by a voting scheme, and developed a methodology to extract small investor sentiment from stock message boards. S Sabherwal et al. (2008) estimated daily abnormal returns using information extracted from TheLion.com stock message board. C Oh (2011) investigated predictive power of stock micro blog sentiment in forecasting future stock price directional movements.

Existing researches related to Internet data analysis application mostly focus on micro-business strategy, while little are used in macroeconomic early warning. The existing studies mostly use Google Insights as their social media information sources, which is a statistic about the search habits of Internet users, reflecting how many users searching a specific keyword within a time period. T Schmidt et al. (2010) used Google Insights data to separately predict Germany private consumption. N Askitas et al. (2009) used Google search data to establish strong correlations between search activities for certain keywords and the unemployment rates in Germany. BD Humphrey (2010) used Google search queries to forecast existing home sales. S Vosen (2011) compared the forecasting results between Google Insights indicator with survey-based indicators.

Our main contributions in this paper are constructing a price sentiment index from large amount of news reports and trying to use this extracted index in macroeconomic early warning.

3 Methodology

3.1 Research Framework/ A Webpage Count Based Sentiment Extraction Approach

To extract meaningful information about inflation from the Internet for macroeconomic early warning, we need to 1) using keywords searching to select webpages relevant to inflation, and label each page with related price sentiment tags, 2) synthesize these pages and tag into an index reflecting inflation, and 3) assess the effectiveness of the index with respect to inflation early warning.

After the first steps work, we can get large amount of labeled webpage. In order to include the text information into macroeconomic early warning, we use the webpage counts related to each key word to transform the unstructured information into quantitative measures. We may constrain the webpage search to certain published time period to get a subset of resulting web pages. The count of webpages in this subset may represent how many people/institute hold similar opinions about a given economic issue. In this way, a series of web page counts may be established to reflect the public opinion variations on the searching keywords. In this paper, we developed a web-based software program Eco Web Fetch to realize the first steps work.

The algorithm processes can be expressed as follows:
price-sentiment-index-construction (N_t, P, N, N_k)
N_t: text samples number
P: positive sentiment tag
N: negative sentiment tag
N_k: searching keyworsds number
Begin:

1. S ← news sample download;
2. S'← sentiment tags labeling;
3. word segmentation;
4. frequency statistics;
5. SK← searching keywords selection;
6. if searching keyworsds number $< N_k$
   go to 1;
7. SK'← searching keywords sentiment tags labeling;
8. P'← webpages searching and counting;
9. PInx← price sentiment index construction;

End.

3.2 Webpage Selection and Sentiment Tags Labeling

Ideally, we may design a web crawler to retrieve every page in the Internet, and select those pages related to consumer prices based on text mining techniques. Practically, existing search engine, such as Google, provides a convenient solution for the webpage selection. Just by typing into a searching keyword in Google, we can get all the webpages related to that keyword, and the resulting webpages may reflect the collective opinions about that search topic. Although there is some chance that the search result may contain irrelevant webpages, the Google search engine has done its best to improve the rank and filter techniques.

Searching Sources Determination. When selecting related webpages, choosing searching sources is important and depends on research purpose. In this paper, Google News search is used to select webpages relevant to price variations in China. We choose Google News as information sources to construct the price sentiment index, instead of Google Insights as previous studies did. Google News represents how many institutions publish news reports relating to that keyword, which is s signal of how the real situation is and is relatively more objective, while Google Insights represent how many users have searched specific keywords within a time period. From users searching habit, when they are interested in one topic, they usually search this specific issue, not use the keywords with obvious sentiment direction. For example when they concern consumer price, typically they usually search with keyword price trend or a specific goods price like eggs price. They rarely use the keywords with sentiment direction like rising price. So the Google Insights results only mean uses concerning about this issue, cannot clearly reflect the user's point of view, while there is always a clear attitude in

a news report. In this research, we expected to use keywords with obvious sentiment direction to search webpage, so we choose Google News as information sources, instead of Google Insights.

Searching Keywords Selection. When using a search engine to select webpages, searching keywords directly determine the quality of searching results. So choosing representative searching keywords becomes a key issue. Searching keywords can be defined by text mining techniques, focusing on word frequency. This process consists of the following steps: firstly, we need text samples from which searching keywords can be extracted. 100 pieces of macroeconomic news reports related to consumer price are downloaded from the Internet. To ensure the text samples are representative, 100 news reports are required to have clear sentiment tags: reflecting high or low consumer price, thus keywords that both identify the positive and negative sentiment can be extracted. 50 high price news reports and 50 low price news reports are selected as text samples. Secondly, word segmentation and frequency statistics (R Feldman, J Sanger, 2006) are applied in each news text. Word frequency is more generalized than semantic-based methods and has been proved to be an effective statistical property to identify text features (B Wuthrich et al., 1998; M Doms, N Morin, 2004). We sort the segmentation results in descending order according to their word frequency, and words that appear most frequently, among the top5 words in each text, are selected as searching keywords. Then we can manually define each words sentiment direction, positive or negative. Generally speaking, choosing 8-10 searching keywords that have obvious sentiment directions is recommended. Too few searching keywords cannot cover all the related news reports while too many keywords may return duplicated information.

Sentiment Tags Labeling. Since we use keywords searching to select relevant webpages, the sentiment of keywords may represent the webpages sentiment. For example, the returned webpage using keyword rising price, we can include them into the class that says price increases. There are sometimes when a news report contains certain keywords such as rising price, the theme of this news report does not mean rising price. But if a keyword appears in the news title, we can say that the theme of this news report is the same with that keyword. So when we use Google search engine, we require that only when a keyword appears in the news title, can we include that webpage into our searching results. Google advanced search also provides function searching for words in the whole page or page title.

Eco Web Fetch can be used to select webpages and label sentiment tags at the same time. Users input several self-defined keywords, specify which classes they belong to, positive or negative, and set keywords weights. The keywords weights are determined by users, either based on experience, expert advice, or text mining statistics like frequency. Moreover, users can identify statistical frequency: weekly, monthly, quarterly or even yearly, and the searching beginning and ending time. By clicking a searching button, Eco Web Fetch can return the

webpage counts related to each corresponding keywords within every time period and frequency that user defined.

3.3 Index Construction

After step 3.2, we can get a group of time-series data, the webpage counts related to each corresponding keywords within every time period. To make the search result comparable across different time periods, we may standardize the index as a ratio of webpage counts of two sets of sentiment keywords of opposite meanings. A price sentiment index constructed in this paper is proposed as:

$$PInx_t = \frac{\sum_{m=1}^{M} w_m \times N_{mpt}}{\sum_{l=1}^{L} w_l \times N_{lnt}} \ . \tag{1}$$

where M and L is the number of positive and negative keywords, w_i (i=m,l) is the i^{th} keyword weight, including both positive and negative, and N_{mpt} is the number of webpage counts related to the m^{th} positive keyword in period t, and N_{lnt} is the number of webpage counts related to the l^{th} negative keyword in period t. We tried various index construction formulas, including the three in Antweiler and Frank (2004), and the results are quite similar, in terms of turning point detection and leading pattern recognition. The ratio transformation is used because of its simplicity and robustness to text sample numbers. Similar with Antweiler and Frank, we did not include neutral keywords because macroeconomic early warning is always committed to detect turning points in economic cycle, while this group provides little information about trend changes.

3.4 Early Warning Performance Assessment

After the price sentiment index is established, how to assess the quality of index in economic early warning is a key problem. In this part, we need to answer questions like: whether price sentiment index leads the CPI data? Or can price sentiment index identify the price inflection points in advance? Econometric methods are used here.

Seasonal Adjustment. There will always be some noise is time series data, especially economic time series, so data preprocess is essential before analyzing and modeling. The seasonal adjustment method (DF Findley, BC Monsell, 1998) is the process of estimating and removing seasonal effects from the original sequence, in order to reflect the basic trends of the series itself more accurately, and further, reflect changes in the economy turning point. Seasonal adjustment method assumes that a time series consists of four components, represented as:

$$Y_t = TC_t + S_t + I_t \ . \tag{2}$$

where Y_t is the original series, TC_t is the trend cycle component, S_t is the seasonal component, and I_t is the irregular component. After seasonal adjustment,

we separate out the respective trend cycle component TC_t from the two series. Any further analyzing and modeling processes are implemented on the TC_t component.

Correlation Analysis. In order to explore the potential correlation between the CPI and the price sentiment index, unit root tests and cointegration test can be conducted on the two TC_t components. Unit root tests are performed respectively on the two series to determine whether they are stationary. Only under two conditions can cointegration test be conducted: the two series are both stationary, or they are both non-stationary but their first order difference series are stationary. The residual-based approach proposed by Engle and Granger (1987) can be adopted to do cointegration test, to further examine whether there are long-term equilibrium relationship between the two series. The test is based on an regression equation:

$$y_{1t} = \beta_2 y_{2t} + \beta_3 y_{3t} + \cdots + \beta_k y_{kt} + \mu_t \ . \tag{3}$$

Where y_1, y_2, $\cdots$, y_k are first-order stationary series. Then estimated residuals can be expressed as:

$$\hat{\mu_t} = y_{1t} - \hat{\beta}_2 y_{2t} - \hat{\beta}_3 y_{3t} - \cdots - \hat{\beta}_k y_{kt} \ . \tag{4}$$

Cointegration test is the process of testing whether residuals series u_t is stationary.

Turning Points Detection. One important application of the constructed price sentiment index is turning points detection, which is the main task of macroeconomic early warning. Since statistical data release delays the real-time news coverage, whether price sentiment index series lead the CPI data, and can price sentiment index identify the price turning points in advance? If the long-term equilibrium relationship between the two is verified, BB algorithm is then used to test whether the price sentiment index can identify the CPI turning points in advance.

The BB algorithm (G Bry, C Boschan, 1971) is proposed to detect turning points identified as local maxima/minima, and turning points satisfying certain censoring rules are defined as the peaks and troughs of individual time series. BB algorithm is based on business cycle theory, where the peak (trough) of the cycle refers to point that is higher (lower) than the points within five months in the series volatility figure.

Through further peak-trough analysis, we can find out if there is a consistent relationship in their turning points by contrasting the peaks and troughs of the two series. If this is confirmed, we can come to the conclusion that price sentiment index can provide significant information for economic early warning.

4 Results

The main goal of this paper is to verify whether the sentiment index extracted from Internet news can detect inflation turning points in advance. A price sentiment index is constructed to represent collective opinions on price, and then evaluated according to its relationship with the CPI. The constructed index time period is chosen from 2002.07 to 2011.08 because this period includes two complete cycle of CPI, with which we can test the reliable relationship between the constructed sentiment index and CPI.

4.1 Eco Web Fetch Searching Results

When using the Eco Web Fetch program, we should firstly define searching keywords. We downloaded 100 inflation related news report from Sina News as our text samples, among which 50 reports are high price news and 50 are low price news. Sample texts are all in Chinese. There are no words but only Chinese Characters in Chinese sentences. So we need to separate each word in the sentence. Therefore, we segmented the continuous Chinese Characters into words and then counted the words frequencies. This work is implemented using a Chinese lexical analyzer ICTCLAS (HP Zhang, HK Yu et al., 2003). After removing stop words, we ranked those segmented words in descending order according to frequency, and found that four positive words increasing price, price increases, rising price, prices rebound and five negative words price dropping, price decreases, falling price, price downward, price declines are always included in the top 10 levels. So we choose the nine words as searching keywords. Taking the text sample in 2011.03 for example, the top 10 keywords and their frequencies are shown in table 1. From the segmentation results in table 1, we can find that there are many neutral words that provide little information about trend changes, like CPI, economy, policy. So we only choose those words that represent clear sentiment (price increasing or decreasing) as our searching keywords.

Using these defined keywords to select web pages in Eco Web Fetch through identifying searching frequency as monthly, we can get the webpage count related to every keyword within each month. Parts of the selected news webpage count are as shown in Table 2. From the table, we can find that in the months when the CPI is high, the total webpage counts of positive keywords are more than that of negative keywords, while searching results are reverse when the CPI is low. The searching results comply with our expectation: reaction of public opinion on consumer price is consistent with real price level.

4.2 Price Sentiment Index Construction

Using the searching results and index constructing function in section 3.3, a time series price sentiment index is constructed. Here each keyword is weighted equally. We considered using weights based on the frequency of individual keywords. From the searching results, we find the webpage counts of some keywords are always larger than others, keyword increasing price for example. This may

Table 1. top 10 keywords frequency for text sample in 2011.03

words	frequency	words	frequency
Consumer Price	456	Central Bank	112
Increasing Price	312	Stable	109
CPI	289	International	104
Rising Price	272	Government	104
Economy	272	Effect	102
Inflation	269	Food	99
Home	265	Development	94
Policy	222	RMB	93
Market	214	Consumption	89
Think	152	Products	86
Price Decreases	146	Real Estate	77
Expect	135	Mobility	73
Country	132	Commodity	72
Pressure	128	Forecast	70
Price Increases	127	Interest Rate	70
Data	123	Price Dropping	70
Regulate and Control	121	Investment	67
Factors	119	Income	67

Table 2. part of Eco Web Fetch searching result

Beginning Time	Ending Time	Positive Class			
		Rising Price	Increasing Price	Price Increases	Prices Rebound
2008/1/1	2008/1/31	44.80	552.00	195.75	4.35
2008/2/1	2008/2/29	31.00	316.40	58.50	4.95
2008/3/1	2008/3/31	44.00	520.00	199.00	3.30
2008/4/1	2008/4/30	39.80	379.20	148.50	4.50
2008/5/1	2008/5/31	36.00	244.40	56.75	4.50
2008/6/1	2008/6/30	36.80	328.40	138.00	5.85
2008/7/1	2008/7/31	46.00	456.00	217.25	6.30
2008/8/1	2008/8/31	37.60	336.40	142.00	5.55
2008/9/1	2008/9/30	31.20	284.00	129.00	5.10
2008/10/1	2008/10/31	25.40	267.20	58.50	4.80
2008/11/1	2008/11/30	24.20	211.20	61.75	5.25
2008/12/1	2008/12/31	27.00	234.40	141.25	7.05

partly prove that those high-frequency words are more important than others. This alternative weighting scheme does not affect any of our conclusion, and so we stick with the simplicity of equal weighting. From the comparison of CPI and price sentiment index in figure 1, we can see that these two series do track each other closely, implying that the extracted sentiment significantly relevant to CPI. The correlation between the CPI and the sentiment time series during the study period is 0.66.

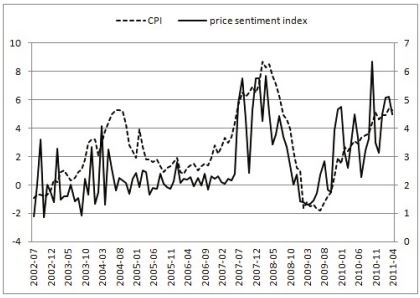
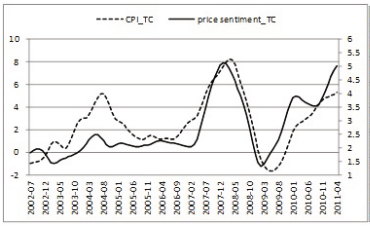


Fig. 1. CPI and extracted price sentiment index

4.3 Correlation Analysis and Turning Point Detection

In order to explore the potential link between the two series further, cointegration test is conducted to examine whether the series move together in a meaningful way over time. Before the cointegration test, unit root tests result shows that the two series are both non-stationary but their first order difference series are stationary. Further, cointegration test can be conducted on the two non-stationary data, adopting the residual-based approach. Cointegration test shows that in 10% significance level, the regression residual loses the non-stationary behavior and is stationary. So we can come to the conclusion that CPI and price sentiment index have long-term equilibrium relationship.

Further BB algorithm is adopted to compute individually the peak and trough points of CPI series and price sentiment index. From figure 2, we can see that after seasonal adjustment, price sentiment index seems leading the CPI series in time. Peak and trough calculating results with BB algorithm are shown in the table in Figure 2. earlier in average. So we can come to the conclusion that price sentiment index does lead CPI series and can identify the price inflection point 2-3 months earlier in average.



Inflection point of price sentiment index	Inflection point of CPI
2004-05 (Peak)	2004-08 (Peak)
2005-08 (Valley)	2006-04 (Valley)
2008-01 (Peak)	2008-03 (Peak)
2009-02 (Valley)	2009-05 (Valley)
Peak average order is 2.5 with standard deviation of 0.71, Valley average order is 5.5 with standard deviation of 3.54	

Fig. 2. Peak-Trough analysis result

5 Conclusion

In this paper, web mining techniques are employed and adapted to extract sentiment information from the Internet news reports. A price sentiment index is proposed in inflation turning points detection to facilitate more timely inflation emergency early warning. Firstly, we collected 100 pieces of macroeconomic news related to consumer price as text samples, from which we select keywords that can reflect positive and negative sentiments in consumer price, using word frequency based text mining techniques. According to words frequency, we found that nine keywords are always included in the top 10 levels. Secondly a web-based software program, Eco Web Fetch, is implemented to search webpages relevant to these nine keywords from Google News, returning corresponding page counts with specific frequency within every time period. Thirdly, a price sentiment index is constructed from these webpage counts, reflecting the aggregated opinions about consumer price in Internet news reports. Evaluation results proved that the proposed price sentiment index can identify the CPI inflection points two to three months in advance. In conclusion, sentiment from social media can provide significant information for inflation emergency early warning. And the webpage count based index construction method is a feasible means to capture comprehensive inflation related dynamics, especially the turning points. Future research on the adoption of proposed method in other macroeconomic aspects will surely increase the timeliness of macroeconomic early warning and forecasting, and support more effective economic emergency management.

Acknowledgement. This research is sponsored by National Science Foundation of China (project number: 71171186, 91224006). The authors would like to thank Professor Cungen Cao for the help of the Eco Web Fetch system design.

References

1. Devitt, A., Ahmad, K.: Sentiment Polarity Identification in Financial News: A Cohesion-based Approach. In: 45th Annual Meeting of the Association of Computational Linguistics, pp. 984–991 (2007)
2. Davis, A.K., Piger, J.M., Sedor, L.M.: Beyond the numbers: An analysis of optimistic and pessimistic language in earnings press releases. Technical report, Federal Reserve Bank of St Louis (2006)
3. Humphrey, B.D.: Forecasting Existing Home Sales using Google Search Engine Queries, `http://econ.duke.edu`
4. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment Classification Using Machine Learning Techniques. In: The Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), Philadelphia, Pennsylvania, pp. 79–86 (2002)
5. Wuthrich, B., Permunetilleke, D., Leung, S., Cho, V., Zhang, J., Lam, W.: Daily Prediction of Major Stock Indices from textual WWW Data. In: 4th International Conference on Knowledge Discovery and Data Mining, pp. 364–368 (1998)
6. Oh, C., Sheng, O.R.L.: Investigating Predictive Power of Stock Micro Blog Sentiment in Forecasting Future Stock Price Directional Movement. In: ICIS (2011)

7. Findley, D.F., Monsell, B.C.: New capabilities and Methods of the X-12-ARIMA Seasonal Adjustment Program. Journal of Business and Economic Statistics (1998)
8. Kaufer, D., Mellon, C.: Flaming: A White Paper, http://www.eudora.com
9. Riloff, E., Wiebe, J.: Learning Extraction Patterns for Subjective Expressions. In: The Conference on Empirical Methods in Natural Language Processing (EMNLP 2003), Sapporo, Japan, pp. 105–112 (2003)
10. Bry, G., Boschan, C.: Cyclical Analysis of Time Series: Selected Procedures and Computer Programs. Columbia University Press for the NBER (1981)
11. Zhang, H.P., Yu, H.K., Xiong, D.Y., Liu, Q.: HHMM-based Chinese lexical analyzer ICTCLAS. In: SIGHAN Workshop (2003)
12. Wiebe, J., Wilson, T., Bell, M.: Identifying Collocations for Recognizing Opinions. In: The ACL 2001 Workshop on Collocation: Computational Extraction, Analysis, and Exploitation, Toulouse, France, pp.24–31 (2001)
13. Doms, M., Morin, N.: Consumer Sentiment, the Economy, and the News Media. FRB of San Francisco Working Paper, No. 2004-09 (2004)
14. Askitas, N., Zimmermann, K.: Google Econometrics and Unemployment Forecasting. Technical report, SSRN 899 (2009)
15. Turney, P.D.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: 40th Annual Meeting on Association for Computational Linguistics (2002)
16. Feldman, R., Sanger, J.: The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press (2006)
17. Engle, R.F., Granger, C.W.J.: Co-integration and Error Correction: Representation, Estimation, and Testing. Journal of the Econometric Society 55, 251–276 (1987)
18. Schumaker, R.P., Chen, H.: Textual Analysis of Stock Market Prediction Using Breaking Financial News: The AZFinText System. ACM Transactions on Information Systems 27(2) (2009)
19. Das, S.R., Chen, M.Y.: Yahoo! for Amazon: Sentiment extraction from small talk on the web. Management Science 53(9), 1375–1388 (2007)
20. Sabherwal, S., Sarkar, S., Zhang, Y.: Online talk: does it matter? Managerial Finance 34(6), 423–436 (2008)
21. Vosen, S., Schmidt, T.: Forecasting private consumption: survey-based indicators vs. Google trends. Journal of Forecasting 30, 565–578 (2011)
22. Schmidt, T., Vosen, S.: A monthly consumption indicator for Germany based on internet search query data. Ruhr Economic Papers No. 208 (2010)
23. Antweiler, W., Frank, M.Z.: Is all that talk just noise? The information content of internet stock message boards. Journal of Finance 59(3), 1259–1295 (2004)
24. Dang, Y., Zhang, Y., Chen, H.: A lexicon-enhanced method for sentiment classification: An experiment on online product reviews. In: IEEE Intelligent Systems, pp. 46–53 (July-August 2010)

Efficient Indexing of the Past, Present and Future Positions of Moving Objects on Road Network

Ying Fang[1], Jiaheng Cao[1], Yuwei Peng[1], Nengcheng Chen[2], and Lin Liu[2]

[1] School of Computer, Wuhan University, China
{fangying,jhcao,ywpeng}@whu.edu.cn
[2] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, China
cnc_dhy@hotmail.com

Abstract. Aim at moving objects on road network, we propose a novel indexing named PPFN*-tree to store past trajectories, present positions, and predict near future positions of moving objects. PPFN*-tree is a hybrid indexing structure which consists of a 2D R*-tree managing the road networks, a set of TB*-tree indexing objects' movement history trajectory along the polylines, and a set of basic HTPR*-tree indexing the position of moving objects after recent update. PPFN*-tree can not only support past trajectory query and present position query, but also support future predictive query. According to the range query time, query in PPFN*-tree can be implemented only in the TB*-tree, or only in the HTPR*-tree, or both of them. Experimental results show that the update performance of the PPFN*-tree is better than that of the PPFI and the R^{PPF}-tree. The query performance of the PPFN*-tree is better than that of the MON-Tree and the PPFI.

Keywords: moving object indexing, PPFN*-tree, TB*-tree, HTPR*-tree, range query, trajectory query.

1 Introduction

Developing efficient index structures is an important research issue for moving object database. Traditional spatial index structures are not appropriate for indexing moving objects because the constantly changing location of objects requires constant updates to the index structures and thus greatly degrades their performance.

Numerous researches are focused on index structures for moving objects. They can be classified into two major categories depending on whether they deal with past information retrieval or future prediction. However, some queries not only involved past positions but also current and future positions of moving objects. Many researchers begin to focus on the index structure of moving object for support querying about the past, the present and the future.

Most of these approaches for indexing moving objects assume free movement of objects in the 2-dimensional space. According to reference [1], applications dealing

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 223–235, 2013.

with moving objects can be grouped into three movement scenarios, namely unconstrained movement, constrained movement, and movement in transportation networks. The latter category is an abstraction of constrained movement, i.e., for cars, one might be only interested in their position with respect to the road network, rather than in absolute coordinates. Then, the movement can be viewed as occurring in a different space than for the first two scenarios, which is called 1.5 dimensional spaces in [2].

For the constrained movement scenario, most index structures are focused on indexing the history trajectories or current position of moving objects. For example, index method proposed by C. S. Jensen *et al.* [3], the Fixed Network R-Tree [4] and MON-tree [5] store the complete trajectories of the moving objects and are capable to answer queries about the past states of the database. However, all of these works are focused on the historical movement and cannot support queries related to the current positions and near future positions of moving objects. IMORS [6] is a hybrid index method for indexing current positions of moving objects on road sectors. It relieves update overheads and provides efficient mechanisms in both searching and updating. However, only current positions of the moving objects can be searched in IMORS.

PPFI [7] is a hybrid indexing structure which consists of a 2D R*-tree built on polylines describing road sectors for managing the fixed networks, a set of 1D R*-Trees indexing objects' movement along the polylines, and a hash structure describing the recent state of moving objects. It is capable of answering query related to the past and current positions of moving objects, and predicting near future positions of moving objects. However, some queries such as spatio-temporal range queries related to position information after the most recent update could not be realized in PPFI.

In this paper, we address the problem of indexing moving objects on road network from the past to the future. We propose a novel indexing method named PPFN*-tree (Past-Present-Future index of Moving Object on Road Network) to store past trajectories, present positions, and predict future positions of moving objects on road network. PPFN*-tree is a hybrid indexing structure that consists of a 2D R*-tree which is built on polylines describing road sectors for managing the road networks, a set of TB*-trees indexing objects' movement history trajectory along the polylines, and a set of basic HTPR*-tree indexing the position of moving objects after recent update. In addition, two hash tables are included in PPFN*-tree. PPFN*-tree can not only support past trajectory and present position query, but also support future predictive query.

The organization of this paper is as follows. Section 2 presents related works and motivations of our work. Section 3 describes data model and update policy in PPFN*-tree, shows the index structure of PPFN*-tree and the corresponding algorithms. Section 4 reports on the performance evaluation. The conclusion is given in section 5.

2 Related Works and Motivations

A number of index structures have been proposed for moving object database. Most of these index structures are focused on free movement of the objects in the 2-dimensional

space. Some index structures only handles past positions or trajectories of moving object focused on free movement in space. The STR-tree [8] attempts to group segments according to their trajectory memberships, also taking spatial locations into account. The TB-tree [8] aims only for trajectory preservation, leaving other spatial properties aside. Based on the MVB-tree, Tao and Papadias propose the MV3R-tree [9] which consists of an MVR-tree and a 3D R-tree to index past trajectory data. Lee Eung Jae *et al.* propose TB* tree [10] for efficiently managing current and past trajectory of moving objects. The proposed method considerably improves updating performance using *auxiliary cache* and reduces index size by removing redundant data for representing MBB. However, all the above indices capture only the positions of objects from some past time up until the time of the most recent update.

Some structures only index the current and near-future positions of moving objects. For example, Tayeb et al. [11] use the PMR-quadtree as their underlying spatial access methods for indexing the future trajectories. Kollios et al. [12] and Papadopoulos et al. [13] use the duality transformation to transform a line segment (e.g., trajectory) from the time-space domain into a point in the two-dimensional space. By introducing parametric bounding rectangles in R-tree, the TPR-tree [14] provides the capability to answer the queries about current positions and future positions. The TPR*-Tree [15] improved upon the TPR-Tree by introducing a new set of penalty functions based on a revised query cost model. Based on the B^+-tree, indices for moving objects not only supporting queries efficiently but also supporting frequent updates are proposed. Jensen et al. propose the B^x-tree [16], which employs space partitioning and data/query transformations to index object positions and has good update performance. ST^2B-tree [17] is a Self-Tunable Spatio-Temporal B+-Tree index for moving object database, which is amenable to tuning. Based on the TPR*-tree, basic HTPR*-tree [18][19] adds creation or update time of moving objects to leaf node entries. Basic HTPR*-tree not only supports predictive queries, but also supports partial history queries involved from the most recent update instant of each object (t^o_{mru}) to the last update time (t_{lu}) of all objects.

Some indexing methods are proposed to support both the past and the future movement of the objects. Sun et al. [20] propose a method for approximate query answering based on multidimensional histograms. The BB^x-tree proposed by Lin et al. [21] retains the old phases so that past, present, and future positions of moving objects are indexed, but it only indexes broken "polylines". Based on the TPR-tree, Pelanis et al. [22] propose the R^{PPF}-tree. It can not only accurately index position for times in-between the most recent instant of each object and the last update time of all objects, but also describe connected trajectories of objects. PPFI*[23] is a hybrid indexing structure which consists of a TB-tree indexing history trajectories from some past time until the time of the most recent position sample of each object $o(t^o_{mru})$, and a HTPR*-tree describing position information since the most recent update instant of each object (t^o_{mru}). It not only supports queries of the positions of moving objects at all points in time, but also supports frequent update.

In order to manage moving objects on road network, some special index structures have been developed. For example, C. S. Jensen et al. [3] proposed an index method which stores the network edges as line segments in a 2D R-Tree and the moving objects

in another 2D R-Tree to index the past trajectories of moving objects in networks. FNR-Tree [4] separates spatial and temporal components of the trajectories and indexes the time intervals that each moving object spends on a given network link. The MON-tree approach [5] further improves the performance of the FNR-tree by representing each edge by multiple line segments instead of just one. However, all of these works are focused on the historical movement of moving objects, and query related to the current positions and near future positions of moving objects could not be realized in these index structures.

IMORS [6] is a hybrid index method for indexing current positions of moving objects on road sectors. It relieves update overheads and provides efficient mechanisms in both searching and updating. However, only current positions of the moving objects can be searched in IMORS. PPFI [7] is a hybrid indexing structure which consists of a 2D R*-tree built on polylines describing road sectors for managing the fixed networks, a set of 1D R*-trees indexing objects' movement along the polylines, and a hash structure describing the recent state of moving objects. It is capable of answering query related to the past and current positions of moving objects, and predicting near future positions of moving objects. Because PPFI uses a hash structure describing the recent state of moving objects, it only supports the prediction of near future positions of moving objects and does not provide predictive range query and predictive nearest neighbor (NN) query.

Our work aims to provide an indexing structure not only to support frequent update of moving objects positions, but also to provide querying from the past to the future for moving objects on road networks.

3 PPFN*-Tree

3.1 Data Modeling and Update Policy

As for object moving on road network, road network model is necessary. In PPFN*-tree, the road network is represented in terms of routes and junctions between the routes, i.e., a network $G=(R, J)$, where R is a set of routes and J is a set of junctions. A route $r \in R$ has an associated polyline $pl = p_0, \ldots, p_n$, where p_i are 2-dimensional points, $0 \leq i \leq n$, and $n+1$ is the size of the route. We term p_0 the start point and p_n the end point of the polyline. The direction of a polyline is from its start point to its end point. A polyline, exemplified in Figure 1, can be described as a sequence of connected line segments.

The measure (m) of a point p located on a polyline is the distance, measured along the polyline, from the start point to point p; see Figure 1.

Function M calculates the measure of any point p located on a polyline pl.

$$M(pl, p) = \begin{cases} 0 & p = p_0 \\ M(pl, p_i) + d(p_i, p) & \exists i (i = \min(\{ j \mid p \in s_j \wedge s_j \in pl) \\ \text{undefined} & \text{otherwise} \end{cases}$$

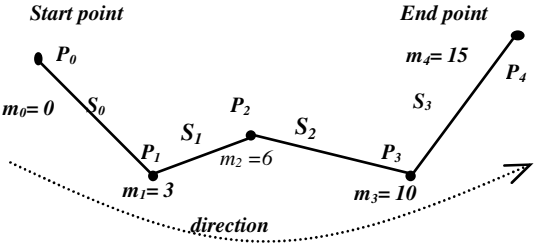


Fig. 1. Polyline

The measure of a point p is equal to the measure of the starting point p_i of segment s_i on which the point p is located, plus the Euclidean distance $d(p_i, p)$.The representation of a moving object on a polyline is a four-tuple $mop = (pl, m, plspd, t)$. Here, $pl \in R$ is the polyline on which the moving object is located; m is the measure giving the moving object's location on *pl; plspd* is the moving object's signed speed along the polyline; and t is the time when the preceding values are valid.

3.2 Index Structure

PPFN*-tree consists of static part and dynamic part. The former contains a 2D R*-tree built on road sectors for managing the road networks and a hash structure H_1. The latter contains a set a TB*-trees indexing objects' movement history trajectory along the polylines, a set of basic HTPR*-tree indexing the position of moving objects after recent update, and a hash table H_2. In order to improve update and query performance, we use H_1 in static part containing entries of the form <*polyid, tree1pt, tree2pt*>, where *polyid* is the polyline identification, *tree1pt* is a pointer to the corresponding TB*-tree, and *tree2pt* is a pointer to the corresponding basic HTPR*-tree.

Figure 2 illustrates the overall data structures of PPFN*-tree. Note R_{road} is a 2D R*-tree for managing the road networks. Each leaf node entry of R_{road} points to a TB*-tree and a basic HTPR*-tree. Note R_{tbi} ($1 \le i \le n$) is a TB*-tree for managing history trajectories of moving objects from some past time until the time of the most recent position sample of each object $o(t^o_{mru})$ moved on polyline S_i ($1 \le i \le n$). R_{hi}($1 \le i \le n$) is the basic HTPR*-tree which indexes position information since the most recent update instant of each object (t^o_{mru}) moving on polyline S_i($1 \le i \le n$).

In $R_{road,}$ leaf nodes contain the information <*mbb, polypt*, *tree3pt, tree4pt*> where *mbb* is the MBB of the polyline, *polypt* points to the real representation of the polyline, *tree3pt* points to TB*-tree of object moved on that polyline, and *tree4pt* points to basic HTPR*-tree of object moving on that polyline. Internal nodes have the following information <*mbb*, *childpt*>, where *mbb* is the MBB that contains all MBBs of the entries in the child node, and *childpt* is a pointer to the child node.

Note R_{tbi} ($1 \le i \le n$) is a TB*-tree for managing history trajectories of moving objects from some past time until the time of the most recent position sample of each object $o(t^o_{mru})$ moved on polyline S_i($1 \le i \le n$). The node structure is as follows:

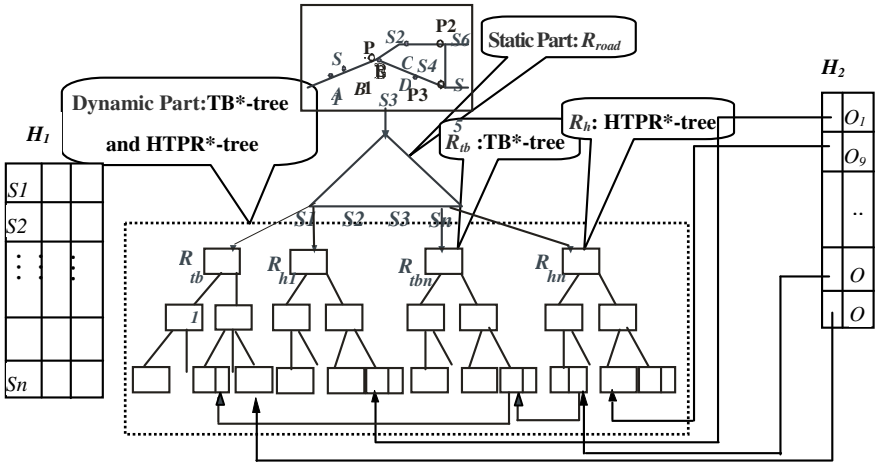


Fig. 2. Overall data structure of PPFN*-tree

$N^{nonleaf}$= [ptr_{parent}, (ptr_1, MBB_1), (ptr_2, MBB_2), …, (ptr_k, MBB_k)]
N^{leaf}= [*Oid*, *polyid*, ptr_{parent}, ptr_{prev}, ptr_{next}, (m_1, t_1), …, (m_k, t_k)]

Where ptr_i ($1 \le i \le k$) points its childe node with MBB_i, and ptr_{parent} points its parent node. Ptr_{parent} is used for improvement of insertion performance by bottom-up update when new data is inserted into index.

In the leaf node, its entry contains not *MBB* but location of moving object. *Oid i*s the identifier of the moving object, and *polyid* is the identifier of the polyline where the object is located. The measure (*m*) of a point *p* located on a polyline is the distance which measured along the polyline from the start point p_0 to point *p*. (m_i, t_i) implies the measure of the objects inside the given polyline are m_i at time t_i. The leaf nodes of TB*-tree are linked with other leaf nodes belonging to the same object. Ptr_{prev} and ptr_{next} are used for managing linked list. Ptr_{prev} is a pointer that points to the last node either in the same R_{tb} or in the different R_{tb}, and ptr_{next} points to the next node either in the same $R_{tb,}$ or in the different R_{tb}, or in the R_h. This information makes it easy to access whole trajectory with minimal cost by visiting an arbitrary leaf node.

R_{hi} is a basic HTPR*-tree to manage position information since the most recent update instant of each object (t^o_{mru}) moving on polyline S_i($1 \le i \le n$). In order to support querying from the past to the future, R_{hi} should be associated with R_{tbj}($1 \le j \le n$) through pointer. We add *ptr* to leaf node entry of R_{hi} which points to a leaf node of R_{tbj} including the most recent history segment. So, the structure of each leaf node entry of R_{hi} is of the form *(oid, tpp, ptr, t*). Here, *tpp*=(*m, v*), with the *m* and *v* being the measure and velocity, respectively, of the object located on polyline S_i at creation or update time *t.* The structure of each non-leaf node entry is in the form of *(ptr, tpbr, st1, st2).* Here *ptr* is a pointer that points to the child node. *St1* is the minimal creation or update time of moving objects included in the child node pointed by *ptr,* and *st2* is the maximum value

compare with *st1*. The *tpbrs* of the Basic HTPR*-tree are bound time-parameterized points which bound objects since time *st1*.

The item in hash table H_2 is defined as vector <*oid, ptr*>, where *oid* denotes the identifier of moving object, and *ptr* denotes physical offset of the leaf node in R_{hi} which object entry locates.

3.3 Insertion

Inserting a new moving object into PPFN*-tree involved R_h, hash table H_1 and H_2. It is carried out by three steps: (1) Based on the polyline identification *polyid* where the object *o* is located, we can get R_{hi} where the object *o* should be inserted through hash structure H_1; (2) A entry (*oid, tpp, ptr, st*) (*ptr* is null) describing the object *o* is inserted in R_{hi} and get the leaf node where entry (*oid, tpp, ptr, st*) is located. (3) The object *o* is registered with its *oid* in H_2 and linked to the leaf node of R_{hi} storing the object *o* with pointer. The detailed algorithm is given in algorithm 1.

Algorithm 1. Insert (*R, o, polyid*)

/*Input: *o* is a moving object with *oid, m, v* and *t*; *R* is the PPFN*-tree, *polyid* is the polyline identification that *o* located*/

1. get the R_{hi} that *o* should be inserted
2. invoke Insert (R_{hi}, *o*)
3. achive the leaf node of *o* stored in R_{hi}
4. register *o* with its *oid* in H_2, link H_2 with leaf node of R_{hi} using *ptr*/*

ENDInsert

3.4 Update

When a moving object *o* with its object identifier reports a new position and velocity to the system, update may be caused by the following three kinds of situation. Firstly, the velocity is changed. Secondly, the allowed position precision threshold is exceeded. Lastly, the new position reported by the object *o* is outside the polyline S_i but inside another polyline S_j, that is the polyline identification *polyid* is changed. In the first two kinds of situation, *o* and *o' (after changed of o)* are located in the same polyline. So, we can get R_{hi} where *o'* will be insert and R_{tbi} where the history segment of *o* will be inserted through H_2, update of PPFN*-tree need only update R_{hi} and insert a history segment to R_{tbi}. Of course, the leaf node entry of the object *o* in R_{hi} can be obtained through H_2 and updated in a bottom-up manner strategy.

Algorithm 2 describes update procedure of PPFN*-tree indexing objects that are moving on the same polyline.

Algorithm 2. Update (*R , o, o', polyid*)

/*Input: *o* is a moving object with *oid, m, v* and *st*; at time *st',* it is changed to *o'; R* is the PPFN*-tree, *polyid* is the polyline identification that *o* located */

1. get the R_{tbi} where history trajectory segment *of o* should be inserted through $R.H_1$
2. get leaf node entry *e1* of *o* in R_{hi} through $R.H_2$
3. invoke R_{hi}.update(*o, o'*)
4. get leaf node entry *e2* of *o'* in R_{hi}
5. get history trajectory segment *e3=(st, st', m, m')*
6. if object *o* has no history trajectories in R_{tbi}
7. insert new segment e3 in R_{tbi} with top-down
8. create pointer of leaf node where *o'* is located in R_{hi} with leaf node where *e3* is located in R_{tbi}
9. else
10. get leaf node *n1*where the last recent history trajectory segment of *o* is located in R_{tbi} through *e1* of *o* in R_{hi}
11. if *n1* has space
12. Insert new segment *e3* in *n1*
13. else
14. create new leaf node *n2* for new segment *e3*
15. find a non-full parent node *n3* of *n1*
16. insert *n2* in tree rooted by *n3* through right-most path
17. create ptr_{prev} pointers of *n2* to *n1* in R_{tbi}, and ptr_{next} pointers of *n2* to entry *e2* of *o'* in R_{hi}

ENDUpdate

3.5 Search Procedure

3.5.1 Spatio-Temporal Range Query

PPFN*-tree can support spatio-temporal range query (given a query window $w=(x_1, y_1, x_2, y_2, t_1, t_2)$, the query is "find all objects that have lain within the area $r=(x_1, y_1, x_2, y_2)$ during the time interval $t=(T_1, T_2)$").

In the range query, when the temporal dimension is zero extent, a special case of range query so-called time-slice query is shown in Figure 3. Figure 4 describes timeslice query and spatio-temporal range query of objects moving on one of polylines on road network.

For the range query, the algorithm receives a spatio-temporal query window *w* and performs in the following three steps. In the first step, a search in the top R*-Tree is performed to find the polylines' MBBs that intersect the spatial query window *r*. Then, the intervals where the polyline intersects *r* are searched using the real polyline

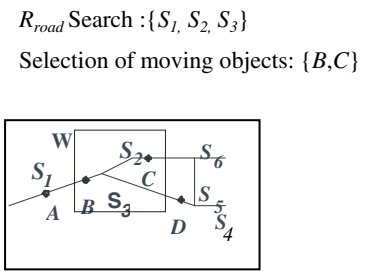


Fig. 3. Time–slice Query of PPFN*-tree

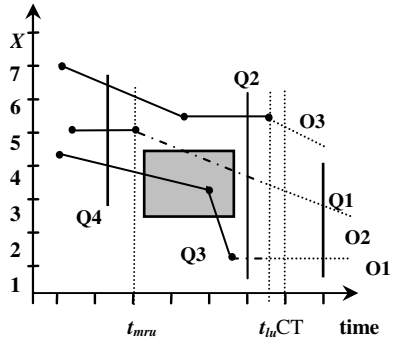


Fig. 4. Querying the Positions of Moving Objects

representation. The result is a set of windows w'= {($M11$, $M12$, T_1, T_2), . . ., ($Mn1$, $Mn2$, T_1, T_2)}, where n is the set size, n ≥ 1, and the interval (T_1, T_2) is the query time interval t. Here, the windows are disjoint and ordered. Finally, given this set of windows w', each w' is performed query in corresponding R_{tbi} or R_{hi} pointed by polyline S_i.

Spatio-temporal range query in PPFN*-tree is bound up with the query time interval t=(T_1, T_2). The detailed search procedure is as follows:

Algorithm 3. RQuery(R, r, T_1, T_2)

/*Input: R is the PPFN*-tree, r is the query spatial area, (T_1, T_2) is the query time interval*/

1. get $R.R_{road}$
2. performing a search in $R.R_{road}$ to find the polylines' MBBs that intersect r
3. for each S_i acquired in step 2
4. change search w to a set of w'= {($M11$, $M12$, T_1, T_2),…, ($Mn1$, $Mn2$, T_1, T_2)} in corresponding polyline S_i
5. if $T2$< $R_{tbi.}$*root.st1*,
6. for each w'=($Mj1$, $Mj2$ T_1, T_2) invoke $R_{tb.}$RQuery$_{.}$($R_{tbi,}$ $Mj1$, $Mj2$, T_1, T_2)
7. else if $T1$>$R_{hi.}$*root. st2*
8. for each ($Mj1$, $Mj2$ T_1, T_2) invoke $R_{h.}$RQuery$_{..}$($R_{hi,}$ $Mj1$, $Mj2$, T_1, T_2)
9. else range query is implemented in both R_{tbi} and R_{hi}

End RQuery(R, r, $t1$, $t2$)

3.5.2 Trajectory Query

Trajectory query is to extract information related to moving objects' trajectories, e.g., "What were the trajectories of trains after they left Wuhan between 5 and 12 today, in the next hour?" We have to (a) select the objects, and (b) select the partial trajectory of each obtained object. In PPFN*-tree, selection of objects can occur by range query and topological query, or obtained by objects identifiers directly. So trajectory query in

PPFN*-tree can be realized by selecting object records in H_2 and getting pointer to extract objects' trajectories during $t=(T_1, T_2)$ in the corresponding R_h and R_{tb}.

3.5.3 Topological Query

Query of the form "find all objects that *enter, leave, cross,* or *bypass* a given area $r=(x_1, y_1, x_2, y_2)$, during the time interval $t=(t_1, t_2)$" is called *topological query* [3].

The *topological query* algorithm consists of three steps: (1) We can obtain polylines intersecting with area r via R_{road}, and area $r=(x_1, y_1, x_2, y_2)$ is changed into $r' = (M_1, M_2)$ in given R_s; (2) Each leaf node entry that (T_1, T_2) intersects with t in every R_s obtained in the first step is examined, and we can obtain the object's measure in t_1 and t_2; (3) The result in step 2 can be used to estimate *topological query*. Figure 4 shows object O1 *leaves* Q3, but O2 *enters* Q3.

4 Performance Study

4.1 Experimental Setting and Details

In this section, we evaluate the performance of the PPFN*-tree with the MON-tree, the PPFI and the R^{PPF}-tree. In all our experiments, we used the network-based moving objects generator proposed in [24]. The generator takes a map of a real road network as input (our experiment is based on the map of Oldenburg including 7035 segments). The positions of the objects are given in two dimensional X-Y coordinates. Since our experiment use edge (polyline) oriented model, we transform X-Y coordinates to the form of (*ployid*; *pos*), where *ployid* denotes the polyline identifier and *pos* denotes the object relative position on the polyline. After that, the total number of polylines is 3803.

4.2 Performance Analysis

- **Update Cost Comparison**

Figure 5 compares the average update cost of the PPFN*-tree, the PPFI and the R^{PPF}-tree as a function of the number of updates. The update cost of the PPFN*-tree increases as the num of updates, but it does not as much as that of the PPFI and the R^{PPF}-tree. This is due to the fact that the HTPR*-tree and TB*-tree of PPFN*-tree adopt bottom-up update strategy to avoid the excessive node accesses, but the PPFI and the R^{PPF}-tree adopt top-down update strategy. Moreover, a history trajectory after the most recent update instant in the R^{PPF}-tree is stored in several entries and even in several nodes, which should be modified when an update occurs. This greatly enhances the cost of update operation of the R^{PPF}-tree. At the same time, a new trajectory inserted into the R^{PPF}-tree also causes *time split* of node, which also reduce the update performance.

- **Query Cost Comparison**

In order to study the deterioration of the indices with time, we measure the performance of the PPFN*-tree, MON-tree and the PPFI using the same query workload.

First, we compare trajectory query performance of the PPFN*-tree with that of PPFI. Figure 6 shows trajectory query performance. We find the PPFN*-tree is more efficient than PPFI under trajectory query, since moving object trajectories were stored in leaf nodes of TB*-tree and basic HTPR*-tree which can be obtained through pointer in H_2 directly. Because the TB*-tree *strictly preserves trajectories* such that a leaf node only contains segments belonging to the same trajectory, trajectory query of the PPFN*-tree will access less leaf nodes than that of the PPFI.

Then, we used range query which tries to find all objects that are moving during a given time interval in a given area. We are interested in the behavior of the indexes according to the follow variables:

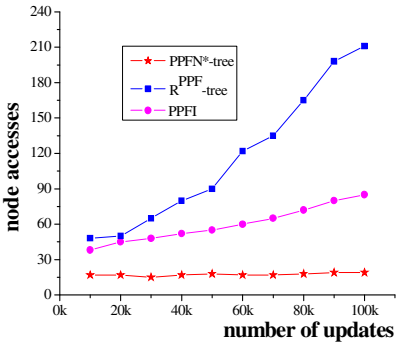


Fig. 5. Update Cost Comparison

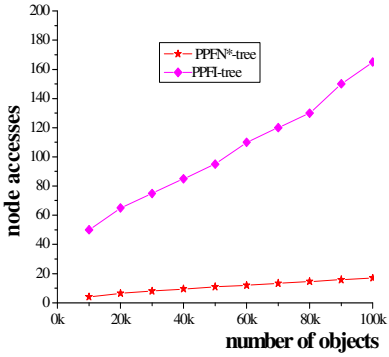
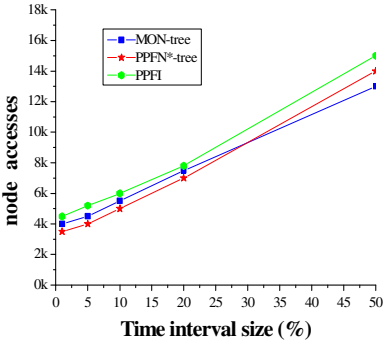


Fig. 6. Trajectory Query Cost Comparison



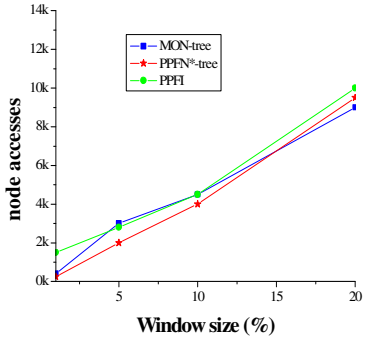


Fig. 7. Range Query Cost Comparison

- Size of the query time interval. We generated queries with a range of 1%, 5%, 10%, 50%, and 100% of the total data set time interval.
- Size of the query window. We generated queries with a range of 1%, 5%, 10%, and 20% of the total data set space.

For each combination of the two variables, we generated randomly 100 queries. The performance of the indexes is then compared by the average number of disk accesses for executing the queries. Figure 7 shows the influence of the query time interval size and the query window size in the performance of queries. As can be seen from these figures, the PPFN*-tree has a linear behavior with respect to the increase of these two variables. For a small range of query interval, the PPFN*-tree shows superior query performance over the MON-tree and the PPFI. However, with the query window size increasing in the largest query time interval size, the query performance deterioration of the PPFN*-tree is more than that of the MON-tree. Because range query of the bottom 2D R-Trees in the MON-tree is superior to that of TB*-tree in the PPFN*-tree. Similarly, with the query time interval size increasing in the largest query window size, the query performance deterioration of the PPFN*-tree is more than that of the MON-tree.

5 Conclusion

In this paper, we develop a novel index structure named PPFN*-tree which consists of a 2D R*-tree which is built on polylines describing road sectors for managing the road networks, a set of TB*-trees indexing objects' movement history trajectory along the polylines, and a set of basic HTPR*-tree indexing the position of moving objects after recent update. PPFN*-tree can not only support past trajectory query and present position query, but also support future predictive query. Extensive experiments prove that the update performance of PPFN*-tree is better than those of the PPFI and the R^{PPF}-tree, trajectory query performance of the PPFN*-tree are significantly improved compared with those of the PPFI. At the same time, for a small range of query interval, the PPFN*-tree shows superior query performance over the MON-tree and the PPFI.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (Grant No.90718027 and Grant No.41171315).

References

[1] Pfoser, D.: Indexing the Trajectories of Moving Objects. IEEE Data Engineering Bulletin 25(2), 2–9 (2002)
[2] Kollios, G., Gunopulos, D., Tsotras, V.J.: On indexing mobile objects. In: Proc. of ACM Symp. on Principles of Database Systems (PODS), pp. 261–272 (1999)
[3] Jensen, C.S., Pfoser, D.: Indexing of network constrained moving objects. In: Proc. of the 11th Intl. Symp. on Advances in Geographic Information Systems (2003)
[4] Frentzos, E.: Indexing objects moving on fixed networks. In: Hadzilacos, T., Manolopoulos, Y., Roddick, J., Theodoridis, Y. (eds.) SSTD 2003. LNCS, vol. 2750, pp. 289–305. Springer, Heidelberg (2003)

[5] Victor, T.D.A., Ralf, H.G.: Indexing the Trajectories of Moving Objects in Networks. GeoInformatica 9(1), 33–60 (2005)
[6] Kim, K.-S., Kim, S.-W., Kim, T.-W.: Fast indexing and updating method for moving objects on road networks. In: Proc. of the 4th Intl. Conf. on Web Information Systems Engineering Workshops, pp. 34–42 (2003)
[7] Fang, Y., Cao, J.: Indexing the Past, Present and Future Positions of Moving Objects on Fixed Networks. In: Intl. Conf on Computer Science and Software Engineering, pp. 524–527 (2008)
[8] Pfoser, D., Jensen, C.S., Theodoridis, Y.: Novel Approaches to the Indexing of Moving Object Trajectories. In: Proc. of the 26th Intl. Conf. on Very Large Databases, pp. 395–406 (2000)
[9] Tao, Y., Papadias, D.: MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries. In: Proceedings of the International Conference on Very Large Databases, VLDB (2001)
[10] Tayeb, J., Ulusoy, O., Wolfson, O.: A Quadtree-Based Dynamic Attribute Indexing Method. The Computer Journal 41(3), 185–200 (1998)
[11] Kollios, G., Gunopulos, D., Tsotras, V.J.: On Indexing Mobile Objects. In: ACM PODS, pp. 261–272 (1999)
[12] Papadopoulos, D., Kollios, G., Gunopulos, D., Tsotras, V.J.: Indexing Mobile Objects on the Plane. In: MDDS, pp. 693–697 (2002)
[13] Saltenis, S., Jensen, C.S., Leutenegger, S.T., Lopez, M.A.: Indexing the Positions of Continuously Moving Objects. In: ACM SIGMOD, pp. 331–342 (2000)
[14] Tao, Y., Papadias, D., Sun, J.: The TPR*-Tree: An Optimized spatiotemporal Access Method for Predictive Queries. In: Proc. of 29th Int. Conf. on Very Large Data Bases, pp. 790–801 (2003)
[15] Jensen, C.S., Lin, D., Ooi, B.C.: Query and Update Efficient B+-Tree Based Indexing of Moving Objects. In: VLDB, pp. 768–779 (2004)
[16] Chen, S., Ooi, B.C., Tan, K.L., Nacismento, M.: ST2B-tree: A Self-Tunable Spatio-Temporal B+-tree Index for Moving Objects. In: ACM SIGMOD, pp. 29–42 (2008)
[17] Fang, Y., Cao, J., Wang, J., Peng, Y., Song, W.: HTPR*-Tree: An Efficient Index for Moving Objects to Support Predictive Query and Partial History Query. In: Wang, L., Jiang, J., Lu, J., Hong, L., Liu, B. (eds.) WAIM 2011. LNCS, vol. 7142, pp. 26–39. Springer, Heidelberg (2012)
[18] Fang, Y., Cao, J., Peng, Y., Chen, N.: Indexing Partial History Trajectory and Future Position of Moving Objects Using HTPR*-Tree. In: Yu, H., Yu, G., Hsu, W., Moon, Y.-S., Unland, R., Yoo, J. (eds.) DASFAA Workshops 2012. LNCS, vol. 7240, pp. 229–242. Springer, Heidelberg (2012)
[19] Sun, J., Papadias, D., Tao, Y., Liu, B.: Querying about the past, the present and the future in spatio-temporal databases. In: ICDE, pp. 202–213 (2004)
[20] Lin, D., Jensen, C.S., Ooi, B.C., Saltenis, S.: Efficient indexing of the historical, present, and future positions of moving objects. In: MDM, pp: 59–66 (2005)
[21] Pelanis, M., Saltenis, S., Jensen, C.S.: Indexing the Past, Present and Anticipated Future Positions of Moving Objects. ACM TODS 31(1), 255–298 (2006)
[22] Fang, Y., Cao, J., Zeng, C., Chen, N.: Indexing the Past, Present and Future Positions of Moving Objects Using PPFI*. In: Proc. of the 8th Intl. Conf. on Networked Computing and Advanced Information Management, pp. 314–320 (2012)
[23] http://www.fh-oow.de/institute/iapg/personen/brinkhoff/generator/

Skyline Query for Location-Based Recommendation in Mobile Application

Linling Ma and Minghua Zhu*

Shanghai Key Laboratory of Trustworthy Computing, Software Engineering Institute, East China Normal University
mhzhu@sei.ecnu.edu.cn

Abstract. The development of mobile computing and communication technology has become more and more important in recent years. In many mobile applications, users need to select "good" information in massive data and skyline query is an approach to achieve the goal in decision-marking situations. In our paper, we present skyline queries in mobile application to select the most qualified parking lots according to the current location and some other information of the parking lots. The information, such as the space number, is updating varying time and the user's location will change, too. Furthermore, we only need to consider the parking lots in local position to gain the optimization goal, rather than global optimization goal. So we need to improve the skyline query and make it appropriate for local query and location-based dynamic query.

Keywords: Skyline Query, Recommendation, Location-based services, Mobile Application, Google Maps API.

1 Introduction

With the development of mobile networks and wireless communication, various mobile applications make it possible for users to access pervasive information from anywhere at any time. Users demand for smart or intelligent services and assist in extracting useful information from massive data. For example, when choosing restaurants in some business districts, users may want a restaurant which is cheaper, closer and has a better review. The query in such scenario involves multiple criteria, such as price, distance and review, and filters all potential "good" or "interesting" data for users in a mobile environment. That is, mobile application needs to provide appropriate recommendation to users.

Skyline operation [1] is a significant operator of location-based services involving spatial data. Furthermore, the data records contain not only the location dimension but also other non-static dimension which will change over time, like the number of remaining seats in a restaurant. Based on such consideration, we should extend the general skyline query for selecting the potential good items.

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 236–247, 2013.

In this paper, we improve the skyline query for our mobile application [2] of parking lot searching. We fetch the nearby parking lot data records and operate the modified skyline query on the distance, price, review, and the number of remaining stalls of the parking lots. The distance will change because of the user's movement and the number of remaining stalls will update with time. Motivated by this fact, we propose a progressive skyline query for dynamic location-based recommendation, which likes SFS (sort filter skyline algorithm), keeps a sliding window to collect skyline points by sorted first in the distance function. In addition, we keep the static skyline points in the sliding window [3], which will not be dropped as the updating of non-spatial dynamic attribute, and coordinate them when the distances change. Hence, we should only concentrate on the non-static skyline points altering with the dynamic attributes. Here we do not choose the more efficient algorithm, such as NN (nearest neighbor algorithm) [4] and BBS (branch-and-bound skyline algorithm) [5], because it spends time building and updating R-tree index with the dynamic distances frequently. Our mobile application visualizes the skyline points in Google Maps and the distance is calculated by parking lots and user's location. The user's location is fixed by GPS in mobile device.

The rest of this paper is organized as follows. In Section 2, we present the preliminaries including the problem statement and the data model and organization. In Section 3, we show a detailed analysis of skyline query processing for the mobile application. In Section 4, we propose implementation of our application and experimental results. The related works are presented in Section 5. Section 6 concludes the paper and puts forward our future work.

2 Preliminaries

2.1 Problem Statement

In the most LBS Mobile Application, users may search some useful information involving more than one aspect. In our work, we develop a mobile application for searching the nearby parking lots with skyline query. The mobile application will provide one or more parking lots which are not worse than the others by skyline query and users can gain a parking lot among the selected ones. A parking lot has some attributes including name, address, longitude, latitude, price, review and the remaining number of stalls.

Towards the attributes of a parking lot, first, we extract the price and review. Second, we fetch the longitude and latitude to indicate the location. Last, the number of remaining stalls is required. Hence, we have a set of data points attributes as (*longitude*, *latitude*, *price*, *review*, *number*), where *number* is the number of remaining stalls. For the distance between the parking lot and user's location, the mobile device can get the current longitude and latitude of the user by GPS and the location of parking lots are already known.

On account of the user's movement, the distance updates with time. All data points pose dynamic skyline queries and the queries involve both distance and

other static and dynamic attributes. In our solution, we first calculate the distance for each data point. Then we filter the data points to content the local distance condition and compute the skyline by (*distance*, *price*, *review*, *number*). When the distances change, we renew filtered data points and skyline points by the skyline query process.

2.2 Data Model and Organization

A parking lot is expressed as a data point, named *DP*. The static attributes of a parking lot is represented by a data point *DPS* in the format of *DPS* (*id*, *name*, *address*, *dlng*, *dlat*, *price*, *review*), where *dlng* and *dlat* mean the longitude and latitude of a parking lot' location. The non-static attributes is number, signed as *DPN*(*id*, *number*). *DPS* is associated with *DPN* by the *id* of a parking lot. A user's location is a query point *QP* with the attributes of longitude and latitude as *QP*(*qlng*, *qlat*), where *qlng* and *qlat* mean the longitude and latitude of a user' location.

In our mobile application, we do not pre-process an index such as R-tree to organize the data records. In contrast, *DP* organized by a list structure in main memory. The list is a point set S = p_1 , p_2, ...,p_N and p_i = (*id*, *distance*, *price*, *review*, *number*) for $0 \leq i \leq N$, where *distance*, *price*, *review* and *number* are the dimensions to be considered in skyline operation. The *distance* here is the distance between two data points on a sphere and it will be calculated at a certain query time. It is calculated as follows:

- $\triangle lng = \frac{dlng}{180} * \pi - \frac{qlng}{180} * \pi$
- $\triangle lat = \frac{dlat}{180} * \pi - \frac{qlat}{180} * \pi$
- $a = \sin(\frac{\triangle lat}{2})^2 + cos(dlat) * \cos(qlat) * \sin(\frac{\triangle lng}{2})^2$
- $distance = R * 2 * \arcsin\sqrt{a}$, (R is the radius of earth)

Mobile users may acquire parking lot information from anywhere at any time. *DPS* should be stored in each mobile device for direct access while *DPN* are placed in the centralized server device, which may update varying over time. When accessing *DPN*, the application will send an HTTP request and wait for the response. Since then, the application will compute skyline points and show the result to users on the Google Map.

3 Skyline Query Processing

3.1 Local Skyline Query

The *skyline* is a set of d-dimensional points, in which the points are superior to any other point on all considered dimensions. The process of selecting the skyline is called *skyline query* [6]. We now give the definition of some related concepts formally. The set of all points is S and each point s ($s \in S$) has d attributes. The d-dimensional attribute space is donated as D.

Definition 1 (Dominance Relationship). *Give two points s^1 and s^2, if s^1 is equal to or better than s^2 in all attributes, and s^1 is better than s^2 at least one attribute, we donate s^1 dominates s^2, and write $s^1 > s^2$. That is:*

$$s^1 > s^2 : \left(\forall a \in D, s^1.a \geq s^2.a\right) \wedge \left(\exists a \in D, s^1.a > s^2.a\right)$$

Definition 2 (Skyline Query). *The skyline query is the process of selecting the skyline, which is the set of points being not dominated by any other point. SK is donated as the skyline. That is:*

$$SK\,(S) = \left\{s^1 \in S | \neg \exists s^2 \in \left(S - \left\{s^1\right\}\right) \wedge s^2 > s^1\right\}$$

We are interested in finding out the local skyline in an matched data points subset of *DP* list according to a given distance range far away from the user's location. The given distance range can be provided by users. For instance, assume there exists five data points in *DP* and the given distance range is 500 meters. The five data points are p_1 = (1, 350, 15, 15, 10), p_2 = (2, 400, 8, 30, 20), p_3 = (3, 600, 10, 5, 5), p_4 = (4, 235,15, 60, 30), p_5 = (5, 500, 15, 15, 30). Intuitively, p_3 is not eligibility because its distance is 600 meters beyond the given range. Data point p_4 dominates p_1 and p_5 while p_4 and p_2 do not dominate each other. So p_2 and p_4 are the local skyline.

3.2 Skyline Query with Dynamic Dimensions

Before the query, we initialize the list S for each p_i = (*id*, *distance*, *price*, *review*, *number*). S is an sequential list ordered by distance dimension. Furthermore, we divide S into two list as S_1 and S_2, where all the data points in S_1 have distance more than the given distance range while S_2 is opposite. For *number* of the *DP*s in S_1, we send an HTTP request to server to obtain by appointed *id*s.

First, we focus on the static dimensions and compute the skyline points of S_1 just involving price and review. These skyline points are denoted as SP_{static} and this lead to Lemma 1.

Lemma 1. *In an specified data points set S and the data points have both dynamic and static dimensions, SP_{static} involving just static dimensions are an subset of the total skyline points involving all dimensions, which denoted as SP_{all}.*

Proof. Obviously the data points in SP_{static} dominate all data points of S in static dimensions. Now if we consider the static dimensions and a dynamic dimension, any data point in SP_{static} will still the member of SP_{all} because they may not be dominated by any data point in S with static dimensions. Thus involving one more dimension will not reject any data point in SP_{static} from SP_{all} and SP_{static} are an subset of SP_{all}.

According the above lemma, we then calculate the SP_{static} and SP_{all} of S_1. The initialization algorithm is presented below:

```
Algorithm. initialization
Input: Q, S
Output: SP_static, SP_all
initialize SP_static and SP_all as null
for each p_i in S
  Compute the distance of p_i and the users location;
  if (distance > given distance)
    add p_i into S1 sequentially;
  else add p_i into S2 sequentially;
for each p_i in S1
  for each p_j in SP_static
    if (p_i is dominated by p_j in static dimensions)
      break;
    if (p_j is the last of SP_static)
      p_i is add into SP_static;
  for each p_k in SP_all
    if (p_i is dominated by p_k in all dimensions)
      break;
    if (p_k is the last of SP_all)
      p_i is add into SP_all;
```

When the number of a data point in S_1 updates in a certain time, SP_{static} will remain while others may change. We denote the changed data point as DP_{change}. There are three situations as follows:

– DP_{change} is in SP_{static}.

Obviously DP_{change} is still in the skyline. If the *number* becomes better, some data points in SP_{all} - SP_{static} will be deleted from the skyline. Because they are dominated by SP_{static} in static dimensions before and now also the dynamic dimension number. If the number becomes worse, some data points in S_1 - SP_{all} may enter the skyline. Because they are dominated by SP_{static} in all dimensions before but now the *number* of them is no longer dominated by DP_{change}.

– DP_{change} is in SP_{all} - SP_{static}.

If the *number* becomes better, the case is the same as the first situation. If the number becomes worse, DP_{change} will be dropped and some data points in S_1 - SP_{all}, which are dominated by DP_{change} before, may enter the skyline.

– DP_{change} is in S_1 - SP_{all}.

In this situation, it is much simpler. If the number becomes better, it may enter the skyline and we should check it. If the number becomes worse, we need to do nothing because no change happened.

The algorithm updateNum is presented below:

```
Algorithm. updateNum
Input: SP_static, SP_all, S1, the list of DP_change
Output: SP_all
for each DP_change in list
  if (DP_change.num become more)
    if (DP_change is in SP_all)
      for each p_i in SP_all-SP_static
        check whether to delete by DP_change
    if (DP_change is in S1-SP_all)
      for each p_i in SP_all
        check whether to delete by DP_change
        check whether to add DP_change into SP_all-SP_static
  if (DPchange.num become less)
    if (DP_change is in SP_all)
      for each p_i in S1- SP_all
        check whether to add into SP_all-SP_static
      if (DP_change is not in SP_static)
        check whether to delete DP_change from SP_all
```

3.3 Location-Based Skyline Query

As we known, the distances are changing with the movement of QP, so the data points in S_1 and S_2 may be restructured. That is, the distances of data points in S_1 may be more than the given distance range and they must be deleted from S_1 and inserted into S_2. Oppositely some data points in S_2 may become the member of S_1. As S_1 and S_2 are ordered list by distance, we just check some data points in the tail end of S_1 and some data points in the head of S_2. The distance of user's movement is donated as dist and and the two situations describe as follows:

- $|dist + distance| > givenDistance$. If the data point in S_1 satisfies the above formula, it may be deleted from S_1.
- $|dist + distance| \leq givenDistance$. If the data point in S_2 satisfies the above formula, it may be inserted into S_1.

All the data points in S_1 now should calculate new distance. When restructuring S_1, we will compute the new skyline points. Here the SP_{static} may change according to the deleted or the inserted data points of S_1. If the deleted data point is in SP_{static}, it will be also deleted from SP_{static}. If an inserted data point is coming, we should compare it with existing SP_{static} to estimate whether it belongs to SP_{static} or not. All the data points in S_1 - SP_{static} should make the skyline operation to get the new skyline.

On the other hand, the data points in S2 now should also be calculated new distance using multi-thread efficiently. The algorithm updateLocation is described as below:

```
Algorithm. updateLocation
Input: SP_static, S2, S1, Q
Output: SP_static, SP_all
recalculate the distance of S2, S1,
sort S2, S1,
for each p_i in S1
  if (p_i.distance > given distance)
    move p_i to S2;
    if (p_i is in SP_static)
      delete p_i from SP_static;
for each p_i in S2
  if (p_i.distance < given distance)
    move p_i to S1;
for each p_i in S1- SP_static
  compute and get the SP_static and SP_all;
```

3.4 Complexity Analysis for the Skyline Query Processing

The skyline query processing is described as the three sections before and now we analyze its complexity in detail.

At first, we define some variables as follows. The size of S, S_1 and S_2 are donated as n, n_1 and n_2. The size of SP_{all} and SP_{static} are donated as m_1 and m_2. The number of DP_{change} is k.

Based on the actual situation, the relationship of these variables can be described as follows: $n > n_2 > n_1 \geq m_1 \geq m_2$ and $k \leq n_1$(k is determined by the real situation). In general, n_2 is far greater than n_1 because we S_1 is the data points list in local range which is relatively very small. If the given distance may not over 1000 meters in our application because it is meaningless if a skyline point is far away from the user.

The complexity of the initialization algorithm is $O(n \log n)$. The description for details is $O(n \log n + n_1 * m_2 + n_1 * m_1)$ and n is far greater than n_1 , m_2 and m_3. The mainly contribution falls on $O(n \log n)$, which is the cost of sorting operation.

Towards the updateNum algorithm, the complexity is $O(k * n_1)$. In this algorithm, the computation is in S_1 and the complexity is not very large because both k and n_1 are far less than n and almost in constant.

As for the updateLocation algorithm, the complexity is $O(n \log n)$ because it need to sort the list again. As we mentioned in above section, the sort are proceeding using multi-thread. So the complexity can reduce to $O(n_1 \log n_1 + n_2 \log n_2)$.

4 Application Implementation and Performance Evaluation

In this section we will describe the implementation of our mobile application and present the evaluations of our prototype. Since it is hard to evaluate the

application by comparison, we display the evaluating result on its own in different aspects. It is an android mobile application based on android 4.0. The *DPS* is stored in the embedded database of the mobile device, SQLite, and the *DPN* is in server. This application was implemented with java programming language in Eclipse IDE.

4.1 Application Implementation Image

Figure 1 shows a image of the application. In a specific user location, application will show the skyline parking lots and if you click a point in the map, it will give a snapshot of the parking lot and you can see the detail information when clicking the snapshot.

4.2 Performance Evaluation

The experiment data are parking lot data in Shanghai, which are collected from internet. One record contains name, address, longitude, latitude, price and review. Towards the number of remain stalls, we generated simulated data every five minutes and send to the server. The application is performed on a mobile phone with these specifications: Android OS 2.2, 576MB RAM, 512ROM and the network environment is local area network.

First, we test the performance of the CPU times. Figure 2 describes the CPU times due to change of the given distance. The given distance is related to the considered data points for local query. As the figure shows, when the *number* updates, the CPU times rise slowly and the longest is still not over 100 microseconds with the increasing of the given distance from 200 meters to 2000 meters. As the *distance* renews, it spends more time on the CPU times for skyline query processing and it still rises slowly. On the other hand, if both of them change, the CPU times are almost the same as the renewing of distance. For these experiment results, we can conclude that our skyline query processing stays relatively stable In terms of time costing, that is, the time will not increase significantly when the data size becomes larger. The CPU time for skyline updating when the number updates is much less than the other aspects because it has no need to restructure the S_1 and SP_{static}.

The next evaluation aims at the HTTP responding times. Here we assume that the application can know whether some numbers have updated and send the HTTP request to get the updated data. When the user's location changes, the HTTP responding times vary little while they are more time-consuming then the above situation. The delay is within the tolerable range and the costs of three aspects are not over 500 microseconds. The results of the evaluation can be seen in Figure 3. So we can draw a conclusion that there are not many points to renew its number in every change. Unlike the first evaluation, there is no big difference on cost between the update of the location and the number. Because for every change, we just send one HTTP request and get the all changed *number* information while the operation to select the information in server takes little time even if the number is high.

Fig. 1. Application implementation image

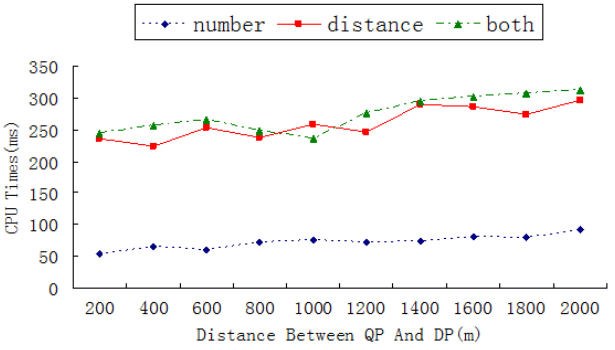


Fig. 2. The CPU times related to change of the given distance

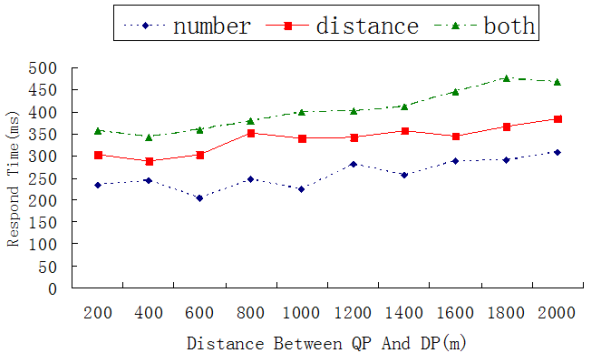


Fig. 3. The HTTP responding times related to change of the given distance

5 Related Work

This section surveys existing work for skyline query processing including general skyline query and dynamic skyline query.

5.1 General Skyline Query

We first provides a brief survey on related work for computing skylines, namely block nested loop (BNL) and sort filter skyline (SFS) [7].

BNL scans all data records one by one. There is a sliding window to keep candidate skyline points and the first data point is inserted into the sliding window. For each point p, there are three cases: (1) If p is dominated by any point in the sliding window, it does not belong to skyline points. (2) If p dominates any point in the sliding window, it is inserted into the sliding window and all points in the sliding window dominated by p are discarded. (3) If p is neither dominated, nor dominates, any point in the sliding window, it is inserted into the sliding window because it may be a member of the skyline. SFS is much similar as BNL. It also keeps a sliding window to collect skyline points. The difference is that all data records are ordered first in some topological sort function. When a data record is inserted into the sliding window, it is exactly a skyline point and can be output to display.

Nearest neighbor (NN) algorithm [5][8] and branch and bound skyline (BBS) algorithm [6] for skyline query are much more efficient than BNL and SFS. They query the data records by R-tree index and find the nearest neighbor point of the pivot point to divide the space. The nearest neighbor point is the skyline point and recursively calls the algorithm in the division space to get all skyline points. The algorithms spend time building and updating R-tree with the dynamic data records frequently, so they are not appropriate for dynamic skyline query.

5.2 Dynamic Skyline Query

For spatial skyline query, Sharifzadeh and Shahabi [9] first introduced the concept of spatial skyline queries (SSQ) and proposed two algorithms, B2S2 and VS2, for static query points and one algorithm, VCS2, for non-static query points changing location over time. Huang and Jensen [10] studied the in-route nearest neighbor skyline queries, issued by users moving along routes towards destinations, with an efficient computation.

Huang, Lu et al. [11] introduced continuous skyline queries for moving objects, where the skyline query point is a moving object and the skyline changes continuously due to the movement of the query point. They assumed that the query point moved in a fixed velocity and each distance between data point and query point changed in some function.

Towards the skyline query on data stream, Tao and Papadias [12] presented two algorithmic frameworks to maintain sliding window skylines on data streams and continuously output the skyline changes.

While the proposed algorithms enable intelligent access to spatial data with static dimension and location-based dimension or the stream data with non-static dimension, our problem should be considered more on spatial data with static dimension, non-static dimension and location-based dimension.

6 Conclusions and Future Work

In this paper, we address the problem of location-based dynamic skyline query processing. Combining with dynamic location information and the real-time updated information, we put forward a query processing to deal with the spatial and dynamic data. The method operates a local skyline operation considering the time spending of obtaining data via HTTP request for mobile application and the practical purpose. It preserves the temporary unchanged skyline points in an sliding window and updates it when necessary. Towards the non-spatial dynamic dimension and location-based dimension, it raises different situations to analysis of possible outcomes in order to avoid unnecessary calculation. Thus we does not need to compute the skyline with every data point.

In next work, We will evaluate our algorithm comparing to the algorithms based on other non-index basic skyline algorithms, such as BNL, DC, LESS and so on.

We intend to use the sensor network technology to obtain the real-time data from the parking lots and research how to update the online data in real-time for mobile application.

Acknowledgments. This work was supported by the Major National Science and Technology Projects of China (No. 2011ZX03005-002) and the Major State Basic Research Development Program of China (973 Program) (NO. 2011CB302902).

References

1. Borzonyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: The 17th International Conference on Data Engineering, pp. 421–430. IEEE Computer Society, Washington, DC (2001)
2. Xiao, Y., Chen, Y.: Efficient distributed skyline queries for mobile applications. Journal of Computer Science and Technology 25(3), 523–536 (2010)
3. Lin, X., Yuan, Y., Wang, W., Lu, H.: Stabbing the sky: efficient skyline computation over sliding windows. In: The 21th International Conference on Data Engineering, pp. 502–513. IEEE Computer Society, Washington, DC (2005)
4. Soudani, N.M., Dastgerdi, A.B.: The Spatial Nearest Neighbor Skyline Queries. International Journal of Database Management Systems 3(4), 65–79 (2011)
5. Papadias, D., Tao, Y., Fu, G., Seeger, G.: An optimal and progressive algorithm for skyline queries. In: The 2003 ACM SIGMOD International Conference on Management of Data, pp. 467–478. ACM, New York (2003)

6. Kodama, K., Iijima, Y., Guo, X., Ishikawa, Y.: Skyline queries based on user locations and preferences for making location-based recommendations. In: The 2009 International Workshop on Location Based Social Networks, pp. 9–16. ACM, New York (2009)
7. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting. In: The 19th International Conference on Data Engineering, pp. 717–816. IEEE Computer Society, Washington, DC (2003)
8. Roussopoulos, N., Kelley, S., Vincent, F.: Nearest neighbor queries. In: The 1995 ACM SIGMOD International Conference on Management of Data, pp. 71–79. ACM, New York (1995)
9. Sharifzadeh, M., Shahabi, C.: The spatial skyline queries. In: The 32nd International Conference on Very Large Data Bases, pp. 751–762. VLDB Endowment (2006)
10. Huang, X., Jensen, C.S.: In-Route Skyline Querying for Location-Based Services. In: Kwon, Y.-J., Bouju, A., Claramunt, C. (eds.) W2GIS 2004. LNCS, vol. 3428, pp. 120–135. Springer, Heidelberg (2005)
11. Huang, Z., Lu, H., Ooi, B.C., Tung, A.K.H.: Continuous Skyline Queries for Moving Objects. IEEE Transactions on Knowledge and Data Engineering 18(12), 1645–1658 (2006)
12. Tao, Y., Papadias, D.: Maintaining Sliding Window Skylines on Data Streams. IEEE Transactions on Knowledge and Data Engineering 18(3), 377–391 (2006)

A Hierarchical 3D Spatial Model Based on User-Defined Data Types

Shulin Cui[1,2,3] and Shuqing Zhang[1,*]

[1] Northeast Institute of Geography and Agroecology, Chinese Academy of Sciences, Changchun 130012, China
slcuifriendly@126.com
[2] University of Chinese Academy of Sciences, Beijing 100049, China
zhangshuqing@neigae.ac.cn
[3] Zhuhai College of Jilin University, Zhuhai 519041, China

Abstract. Based on user-defined data types, a novel 3D data model is proposed in this paper. The spatial data model is a hierarchical structure consisting of Objects, Sub-objects, and Geometries. Objects are composed of Sub-objects, which in turn are made up of Geometries. The relationship between an object and its sub-objects is many to many in the data model. A geometry object can be represented by three different types: discrete object, function and object reference. We first describe spatial data types using generic terminology without referring to any specific solution. And then discuss how to structure spatial data into tables and define the right table structure. As the rules of geometry object construction do not concentrate on triangles or planar polygons, this model is able to handle complex objects which are an essential functionality for building large-scale cyber cities.

Keywords: User-defined data type, Sub-object.

1 Introduction

Topographic objects are getting more complex due to modern land use, such as high-rise buildings, tunnels, underground shopping malls and utility networks [1-2]. The technological progress of both spatial data acquisition (GPS, TLS, etc.) and computer tools (hardware, software, Internet, etc.) have enabled the creation of mass-scale urban models in three-dimensional form [3-4]. However, suitable data models for mass-scale data storage, analysis and efficient query handing are missing.

In literature, many geometry primitive for modeling 3D objects have been proposed [5]. A simple 3D object can be represented basically in three different ways as a polyhedron, triangulated polyhedron or tetrahedron. All the three representations have advantages and disadvantages [6]. While the tetrahedron is widely used in modeling geological formations [7], the polyhedron is the most appropriate representation

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 248–256, 2013.

for man-made objects, such as buildings, bridges, tunnels, etc. [8]. Triangulated polyhedron is compromise between the two ensuring at least the simplicity of the polygons [5].

Geo-DBMSs make it possible to manage large spatial data sets in data bases [8]. Presently, most mainstream DBMSs, such as Oracle, IBM DB, Informix, and Ingres, have already implemented spatial data types and spatial functions usually in an object-relational spatial extend to RDBMS [5]. Moreover, some of them have supported the creation and storage of three-dimensional geometry objects. For example, Oracle Spatial supports the storage and retrieval of three-dimensional spatial data, which can include points, point clouds (collections of points), lines, polygons, surfaces, and solids [9].

But the real world objects seem to be not limited only to this simple geometry types. Moreover, the simple 3D volumetric data type would be still insufficient for modeling more real-world phenomena. Thus, these supported geometry types in current DBMSs are still limited and more 3D geometry types especially for volumetric objects are needed [11]. However, a complex geometry data type and corresponding 3D spatial operations are missing in all DBMS.

The ISO/TC211 spatial schema [11] provides conceptual schemas for describing and manipulating the spatial characteristics of geographic features. It should be noticed that the Abstract Specifications discuss a wide range primitives, such as cone, sphere, triangulated surfaces and freeform shapes including Bézier, B-spline, Cubic-spline, Polynomial spline and NURBS etc. However, it is outside the scope of this standard to specify the interface of the 3D spatial objects within the context of a DBMS [9]. Although freeform shapes can be simulated by tiny simple elements, it is quite unrealistic and inefficient to store all these elements into a DBMS, especially when shapes are rather huge or complex [12].

The rest of this paper is organized as follows. In Section 2, the concept model and logical model are described in detail. In Section 3 an example of the construction of 3D spatial objects by our model is introduced. In Section 4 an implementation specification in object-relational DBMS is given. Finally, Section 5 presents the conclusion.

2 Data Model

2.1 The Conceptual Model

The spatial data model is a hierarchical structure consisting of Objects, Sub-objects, and Geometries (Fig. 1). Objects are composed of Sub-objects, which in turn are made up of Geometries. The relationship between an object and its sub-objects is many to many in the data model. A geometry object can be represented by three different types: Disc_Geom, Function and Obj_ref.

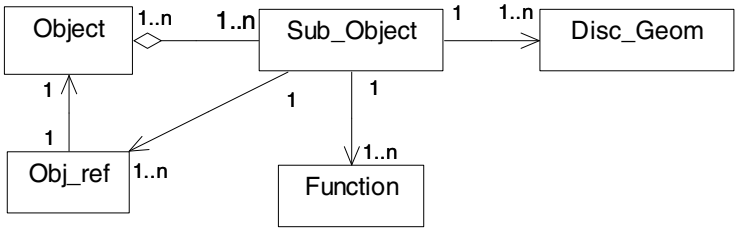


Fig. 1. The conception model

(1) Sub-object

A spatial object may be dissected into a number of smaller independent objects. These smaller objects are then called Sub-objects. Sub-object can be a point, sub-line, sub-surface, or sub-body for basic shape type: multi-point, line, surface and body respectively.

(2) Function

Sub-object is modelled by a function as follows,

Function ：= {functionName, parameter}

Where functionName is the name of the function; parameter is the set of parameters, in which variables are also included.

(3) Disc_Geom

A Disc_Geom type geometry is the representation of a spatial feature, modeled as an ordered set of primitive elements. An element is the basic building block of a geometry object. The spatial element types are point, line strings and polygons. A geometry object can consist of a single element, or a homogeneous or heterogeneous collection of elements.

(4) Obj_ref

In the process of data generation, if a new object is made by referring to the prototype of a ready-made object, the object is called Obj_ref type object. The Obj_ref type is defined as following,

Obj_ref ：= {offset, altitude }

Where offset denotes the coordinate offset of the local coordinate system of the new object or sub-object with respect to the global coordinate system; altitude denotes the transformation relationship between the two coordinate systems.

2.2 The Logical Model

In some cases, the spatial database design can be a complex activity. In general approach, different kinds of geometries have to be stored in separate table columns. In order to overcome the disadvantage, they are stored in one or multiple tables as sub-object type.

Based on the conceptual model above, the logical model is given in Fig. 2. Objects are stored in the table obj_tab. There are four attributes in the table obj_tab, including obj_id, obj_mbb, sub_obj_tab_name and sub_obj_id_set. obj_id is the primary key

for the table. Each sub-object can be stored in the table temp_obj_tab, fun_obj_tab or disc_obj_tab. The sub-object stored in the table temp_obj_tab refers to a row of the table obj_tab.

3 An Example of the Construction of 3D Spatial Objects

An example of modeling a simple building illustrates how to store geometry data using user-defined data types (Figs. 3-4). The object named Building 1 is stored in the first

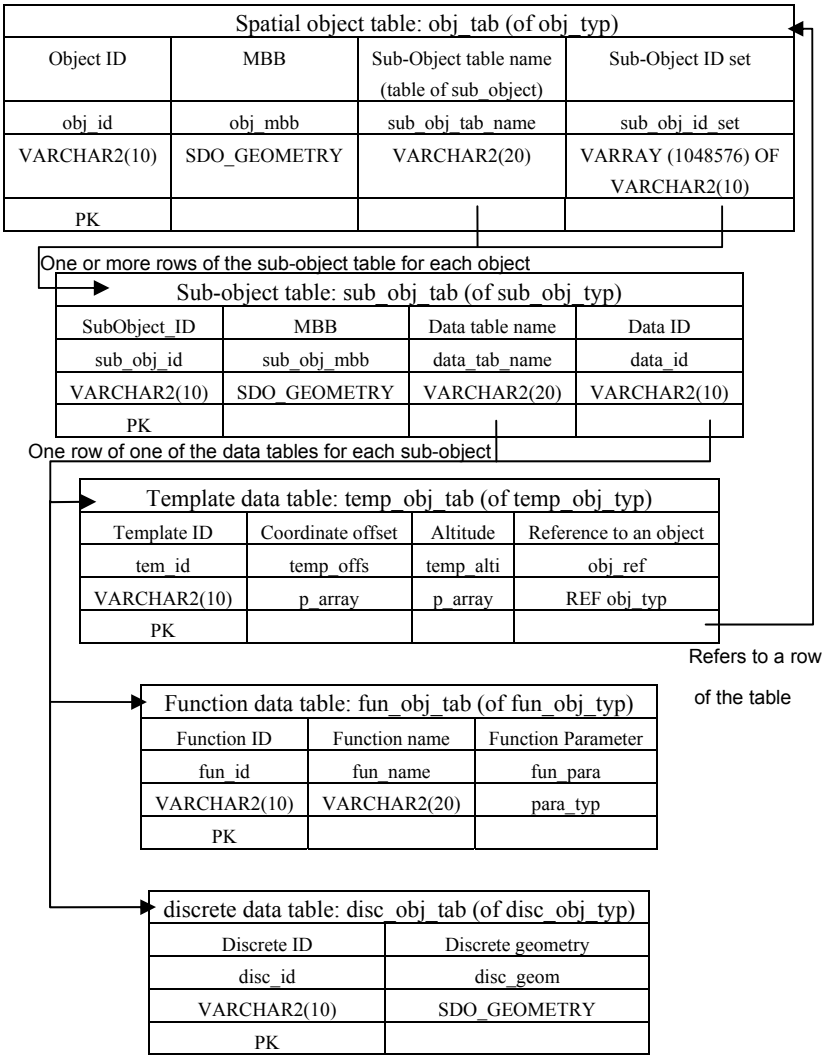


Fig. 2. The logical model

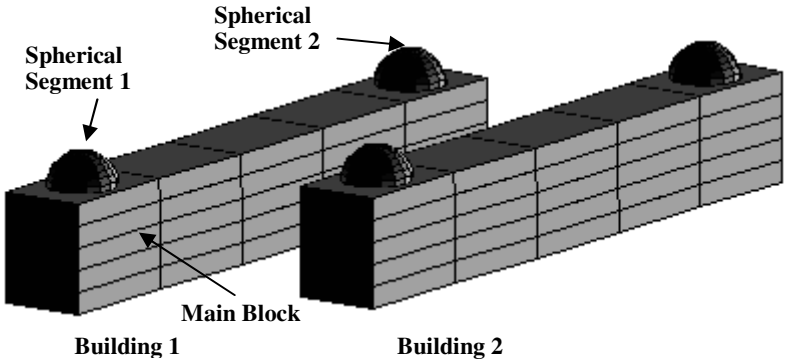


Fig. 3. A simple building

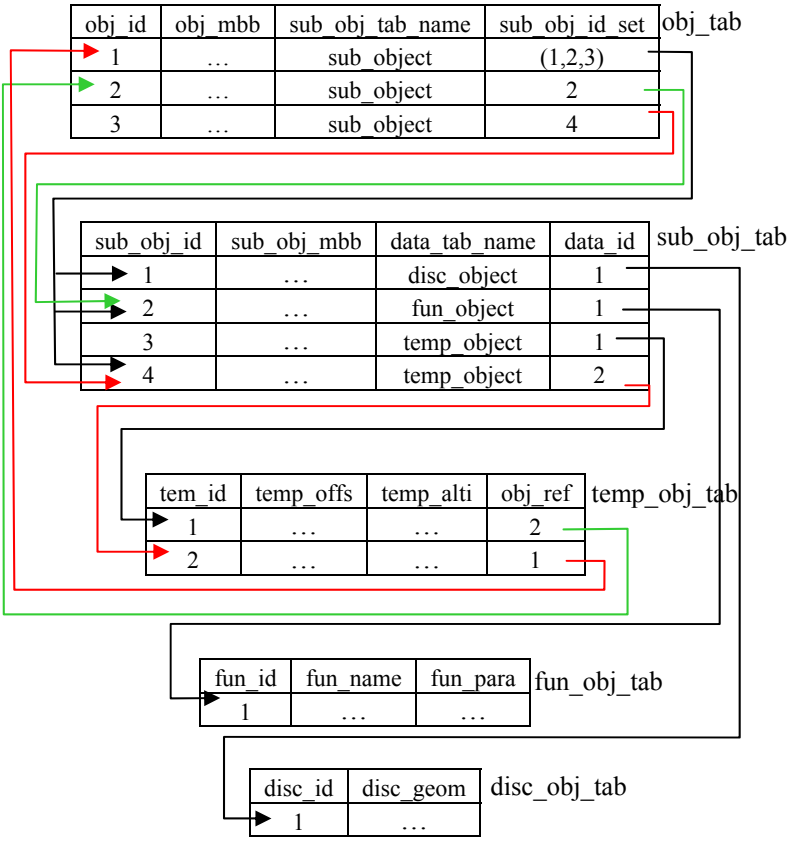


Fig. 4. The building stored in an object-relation database

row of the table obj_tab, and includes three sub-objects whose values in the fieldsub_obj_id are 1, 2 and 3. The first sub-object named Main Block is stored in the first row of the table disc_obj_tab. The second sub-object named Spherical Segment 1 is stored in the first row of the table fun_obj_tab. The third sub-object named Spherical Segment 2 is an Obj-ref type object stored in the first row of the table temp_obj_tab, which refers to the second object in the table obj_tab. The second object in the table obj_tab is also the second sub-object named Spherical Segment 1. The third object in the table obj_table includes one sub-object referring to the first object in the table obj_tab.

4 Implementation

For flexible implementation of the data model an object-relational database was selected (Oracle 11g). We create used-defined data types to represent new type objects. Object references are used to express some of the relationships among them. Moreover, Collection types varrays is used to model multi-valued attributes.

The following statement defines an array type for list of the number of sub-object.

```
CREATE TYPE id_set AS VARRAY (1048576) of VARCHAR2(10);
```

The following statement defines the object type obj_typ to represent spatial object, which uses id_set type as building blocks. A SDO_GEOMETRY object sub_obj_mbb is used to represent the maximal bounding boxing of the spatial object.

```
CREATE OR REPLACE TYPE obj_typ AS OBJECT (
obj_id VARCHAR2(10),
obj_mbb SDO_GEOMETRY,
sub_obj_tab_name VARCHAR2(20),
sub_obj_id_set id_set
);
```

In an object table, data is stored in the structure defined by the table's type, making it possible to retrieve an entire, multilevel structure of data with a very simple query. The following statement defines an object table obj_table to hold objects of type obj_typ.

```
CREATE TABLE obj_tab OF obj_typ (
PRIMARY KEY (obj_id));
```

The following statement defines the object type sub_obj_typ to represent sub-object.

```
CREATE OR REPLACE TYPE sub_obj_typ AS OBJECT (
sub_obj_id VARCHAR2(10),
sub_obj_mbb SDO_GEOMETRY,
data_tab_name VARCHAR2(20),
data_id VARCHAR2(10)
);
```

The following statement defines an object table sub_obj_tab to hold objects of type sub_obj_typ.

```
CREATE TABLE sub_obj_tab OF sub_obj_typ (
PRIMARY KEY (sub_obj_id));
```

The p_array type is defined as follwos:

```
CREATE TYPE p_array AS VARRAY (1048576) OF NUMBER;
```

The type temp_obj_typ refers to obj_typ by containing an attribute that is a REF to objects of obj_typ. Instances of type temp_obj_typ are objects that represent object references to the corresponding Template type objects.

```
CREATE OR REPLACE TYPE temp_obj_typ AS OBJECT (
temp_id VARCHAR2(10),
temp_offs p_array,
temp_alti p_array,
obj_ref REF obj_typ);
```

The following statement defines an object table temp_obj_tab to hold objects of type temp_obj_typ.

```
CREATE TABLE temp_obj_tab OF temp_obj_typ (
PRIMARY KEY (temp_ id));
```

The following statement defines the object type para_typ to represent the set of parameters in function type object.

```
CREATE TYPE para_typ AS OBJECT (
uknot_count NUMBER,
uknot p_array,
vknot_count NUMBER,
vknot p_array,
u_stride NUMBER,s
v_stride NUMBER,
ctlarray p_array,
uorder NUMBER,
vorder NUMBER
);
```

The following statement defines the object type fun_obj_typ to represent Function type object.

```
CREATE TYPE fun_obj_typ AS OBJECT (
fun_id VARCHAR2(10),
fun_name VARCHAR2(20),
fun_para para_typ);
```

The following statement defines an object table fun_obj_tab to hold objects of type fun_obj_typ.

```
CREATE TABLE fun_obj_tab OF fun_obj_typ (
PRIMARY KEY (fun_id));
```

The following statement defines the object type disc_obj_typ to represent Disc_Geom type object.

```
CREATE TYPE disc_obj_typ AS OBJECT (
disc_id VARCHAR2(10),
disc_geom SDO_GEOMETRY);
```

The following statement defines an object table disc_obj_tab to hold objects of type disc_obj_typ.

```
CREATE TABLE disc_obj_tab OF disc_obj_typ (
PRIMARY KEY (disc_id));
```

5 Conclusion

Traditional 3D spatial data models suffer from problems such as data redundancy, deficiency in data representation, data analysis, visualization, etc. Our hierarchical spatial model established on the basis of complex and user-defined types can effectively represent any complex object. Moreover, this paper provided a solution to spatial information management, and how it can be implemented in an object-relational database. In general, different kinds of geometries have to be stored in separate table columns. This may lead to a practical inconvenience. In our model different geometry types are stored in the same table column. Users can define tables containing columns sub_obj_ab_name and sub_obj_id_set to store the name of sub-object table and the number of sub-object.

Acknowledgements. This research was financially supported by the auspices of the National High Technology Research and Development Program of China (863 Program) (No. 2011AA120300 and No. 2011AA120302), the Key Research Program of the Chinese Academy of Sciences (No. KZZD-EW-07-02-003).

References

1. Penninga, F., Van Oosterom, P.J.M.: A Simplicial Complex-based DBMS Approach to 3D Topographic Data Modelling. International Journal of Geographical Information Science 22, 751–779 (2008)
2. Cui, S.L., et al.: Point-In-Polyhedra Test with Direct Handling of Degeneracies. Geo-spatial Information Science 14, 91–97 (2011)

3. Yoo, B.: Rapid Three-dimensional Uurban Model Production Using Bilayered Displacement Mapping. International Journal of Geographical Information Science 27, 24–46 (2013)
4. Koussa, C., Koehl, M.: A Simplified Geometric And Topological Modeling Of 3D Building Enriched By Semantic Data: Combination Of Surface-Based And Solid-Based Representations. In: ASPRS 2009 Annual Conference Baltimore, Maryland (2009)
5. Zlatanova, S.: 3D Geometries in Spatial DBMS. In: 3D-GIS, pp. 1–14. Springer (2006)
6. Zlatanova, S., Holweg, D., Coors, V.: Geometrical and Topological Models for Real-time GIS. In: Proceedings of UDMS 2004, pp. 27–29 (2004)
7. Breunig, M., Zlatanova, S.: 3D Geo-DBMS. In: Zlatanova, S., Prosperi, D. (eds.) Large-scale 3D Data Integration – Challenges and Opportunities, pp. 87–116. Taylor & Francis, Boca Raton (2006)
8. Arens, C., Stoter, J.E., Van Oosterom, P.J.M.: Modelling 3D Spatial Objects in a Geo-DBMS Using a 3D primitive. Computers & Geosciences 31, 65–177 (2005)
9. Oracle Spatial: Oracle Spatial Developer's Guide 11g Release 2 (11.2). E11830-06 (2010)
10. Azri, N.S., Rahman, A.A.: Modelling of Primitive Volumetric Objects in Geo-DBMS. In: Map Asia 2010 & ISG 2010 (2010)
11. ISO/TC 211: Geographic information-Spatial schema. ISO 19107 (2003)
12. Pu, S.: Managing Freeform Curves and Surfaces in a Spatial DBMS. MSc Thesis, TU Delft (2005)

Data Interpolating over RFID Data Streams for Missed Readings

Yingyuan Xiao[1], Tao Jiang[1], Yukun Li[1], and Guangquan Xu[2]

[1] Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Key Laboratory of Computer Vision and System, Tianjin University of Technology, 300384, China
[2] School of Computer Science and Technology, Tianjin University, 300072, Tianjin, China

Abstract. While tracing objects or analyzing human activities with RFID data sets, the quality of RFID data is a crucial aspect. The raw RFID data streams, however, tend to be noisy, including missed readings and unreliable readings. Traditional data cleaning tends to focus on a small set of well-defined tasks, including transformation, matching, and duplicate elimination. In this paper, we focus on exploring efficient methods for interpolating missed readings. We propose a novel probabilistic interpolating method and three novel deterministic interpolating methods based on time interval, containment relationship and inertia of objects, respectively. We conduct extensive experiments and the experimental results demonstrate the feasibility and effectiveness of our methods.

Keywords: l RFID data stream, missed readings, data cleaning, data interpolating method.

1 Introduction

Radio Frequency Identification (RFID) has been deployed in many areas such as supply chain management, intelligent buildings, and retails [1]. RFID readers monitor their surrounding RFID tags and produce data that is then processed, aggregated and queried by the applications. One of the main challenges for these applications is the unreliability of the data by the RFID readers, which limits the widespread adoption of RFID technology. Generally, the unreliability of RFID data streams is caused by missed readings and imprecise readings. For example, RFID readers often capture only 60-70% of the tags in their vicinity [2]. To reduce the effects of these errors, the raw data collected from RFID readers must be appropriately cleaned before using them. Existing applications tend to use tedious post-processing and application-specific means to clean RFID data. In contrast, we consider over separating cleaning from application logic by interposing a data cleaning tier between RFID readers and applications. In this paper, we explore efficient methods for interpolating missed readings. The main contributions of this work include:

1) We propose three novel deterministic interpolating methods based on time interval, containment relationship and inertia of objects, respectively.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 257–265, 2013.

2) We propose a novel probabilistic interpolating method based on normal distribution.

3) We conduct extensive experiments that demonstrate the effectiveness of our proposed approaches over existing solutions.

The rest of the paper is organized as follows. Section 2 reviews the related work on RFID data cleaning. Three deterministic interpolating methods and a probabilistic interpolating method for processing missed readings are proposed in Section 3. Extensive experiments and evaluations are reported in Section 4. We conclude this paper in Section 5.

2 Related Work

The unreliability of RFID data has been widely studied. Many data cleaning techniques have been proposed to improve the quality of RFID data collected from noisy environments [3-12]. Traditional data cleaning tends to focus on a small set of well-defined tasks, including transformation, matching, and duplicate elimination [3, 4].

An important work on data cleaning is to interpolate the dropped readings. To compensate for the inherent unreliability of RFID data streams, one common method is smoothing filter [5, 6, 7, 8]. All missed readings will be filled up if there is at least one reading of the same object within the smoothing window. Most RFID middleware systems employ the smoothing filter, a sliding-window aggregate that interpolates for lost readings. The first declarative, adaptive smoothing filter (called SMURF) for RFID data cleaning is proposed in [8]. SMURF models the unreliability of RFID readings by viewing RFID streams as a statistical sample of tags in the physical world. According to the statistical sample method, all the lost readings will be inserted, if the read rate of readings within the time window above a threshold. The probabilistic model based on confidence or particle filters is proposed in [9]. The probabilistic model is based on the training results related to history data, so results after interpolating lost readings are not accurate enough. Different from the above methods, some works focus on cleaning data with specific application semantics by using rule-based integrity constrains [10, 11]. Chen et al. [12] take full advantage of data redundancy to clean RFID data streams. In addition, Jiang et al. [13] explore to use communication information for RFID data cleaning and make RFID readers produce less dirty data at the early stage.

3 The Proposed Methods for Data Interpolating

In this section, we propose two kinds of methods for interpolating missed readings, i.e., deterministic interpolating method and probabilistic interpolating method.

3.1 Deterministic Interpolating Methods

For the general missed readings, we propose three deterministic interpolating methods (i.e., time interval-based method, containment relationship-based method and inertia-based method).

3.1.1 Time Interval-Based Method

RFID technology is usually error-prone: tags that exist are frequently missed while other tags that are not in a reader's normal view are sometimes read. For example, in a retail store, a bottle of beer is sighted by reader R_i at time T_{i-1}, and later detected by the same reader R_i at time T_{i+1}, but not observed by any reader at time T_i.

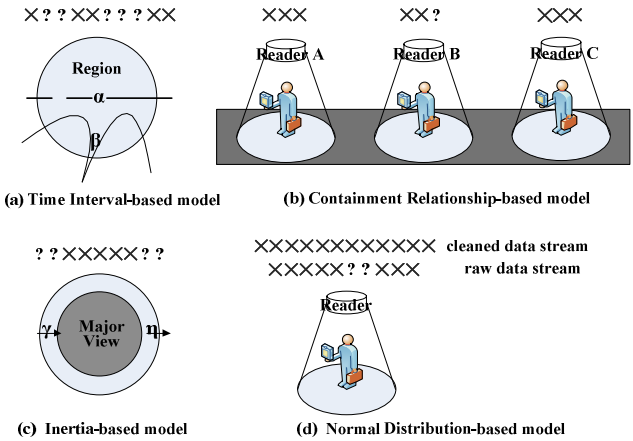


Fig. 1. The situations of missed readings

Fig.1(a) illustrates the situation. In Fig.1(a), the big circle represents detection region of a reader, the lines labeled by α and β denote the routes while objects are going through the normal view of a reader, the gap of a line represents lack of picture, '×' denotes one reading of a tag, and '?' represents lack of a reading. The criterion, to determine whether one tag appearing in the detection region of a reader or not, is the *acceptable time interval*, which is denoted as T_θ. T_θ means the largest possible time interval at which a tag stays in a reader's normal view. The introduction of T_θ is based on the intuition that objects don't usually disappear from the detection region immediately once they come into it. Thus, if there are a missed reading of one object in a reader's normal view at the time instance T_i, the proposed time interval-based method firstly computes the time interval $T_{interval}=T_i - T_{last}$, where T_{last} denotes the recent time when the object was captured by the reader before T_i, and then compares with the *acceptable time interval* T_θ. If $T_{interval}$ is less than or approximately equal to T_θ, it will interpolate the missed reading. Otherwise, it handles with the probabilistic interpolating method, which will be described in the subsequent subsection.

The processing algorithm of time interval-based method is described as follows:

Algorithm 1. Time Interval-based Interpolating

Input: raw RFID data streams and *acceptable time interval* T_θ
Output: the cleaned data streams

1. ***for*** the readings of each object *o* captured by a reader
2. ***if*** the reading of *o* at time T_i is lack
3. $T_{interval}=T_i-T_{last}$;
4. ***if*** $T_{interval} < T_\theta$ or $T_{interval} \approx T_\theta$
5. Insert the missed readings;
6. ***else***
7. Call the probabilistic interpolating method;
8. ***for*** the readings of each object captured by two different readers
9. Call the probabilistic interpolating method;

3.1.2 Containment Relationship-Based Method

In supply chain management and monitoring systems, readings of each object are usually not continuous. The situation is depicted in Fig.1 (b). Note that A, B and C are adjacent readers fixed at an aisle. The following is an application scenario based on Fig.1 (b).

Example 1: Alice takes a purse which contains a bunch of keys with tags, and is going through the aisle. These keys are detected by reader R_{i-1} at time T_{i-1} and reader R_{i+1} at time T_{i+1}, but some of them are not captured by reader R_i at time T_i.

To handle the dropped readings, we may utilize the containment relationship of objects (e.g., the abovementioned keys have containment relationship, because they are contained in the same purse as a whole). For the objects having the containment relationship, if some of these objects are detected by a reader, then we can infer the other objects must also appear in the normal view of the reader. The containment relationship-based method adopts the above idea. Specifically, the processing algorithm of containment relationship-based method is formalized as follows:

Algorithm 2. Containment relationship-based interpolating

Input: raw RFID data streams and the containment relationship among objects
Output: the cleaned data streams

1. ***for*** the readings of each object *o* captured by a reader
2. ***if*** the reading of *o* at time T_i is lack while the reading of the other object, which has the containment relationship with *o*, is captured at time T_i
3. Insert the missed reading for the reader at T_i;
4. ***else***
5. Call the time interval-based method;

3.1.3 Inertia-Based Method

When an object comes into or goes out of the minor detection region of one reader, missed readings usually happen due to the low read rate in this region. Fig. 1(c) shows the situation. The dark color region is the major view of a reader, and light color one is the minor view of the reader, and the arrows notated by γ and η denote the actions of entering and exiting the minor view respectively. The following is an example base on the situation.

Example 2: A forklift picks up an icebox passing through an entrance, and then comes into a warehouse. The icebox is captured at time T_{i-1} by reader R_{i-1} fixed at the entrance, at time T_i by reader R_i fixed in the warehouse, and later are not captured any more.

Entering or exiting a region is a consecutive action which has the property of keeping former state (i.e., inertia). Based on the idea of inertia, we can insert some readings before the first reading of an object and after the last reading of the object, if one reader has fewer readings. The processing algorithm of inertia-based method is described as follows:

Algorithm 3. Inertia-based interpolating

Input: raw RFID data streams and inertia intensity of objects
Output: the cleaned data streams

1. ***for*** the readings of each object *o* captured by a reader
2. ***if*** the number of the readings is fewer than the number required by inertia intensity
3. Insert some readings before the first reading and after the last reading, respectively;
4. ***else***
5. Call the time interval-based method;

3.2 Probabilistic Interpolating Method

The whole detection range of a reader can be partitioned into major detection region and minor detection region. The read rate is higher in the major detection region while is lower in the minor detection region. Generally, the raw readings comply with normal distribution whose parameters are mean value μ, variance σ^2 and independent variable *t*. Here, mean value μ is the middle time when one object is going through the detection range, variance σ^2 is the deviation of the time of readings, and independent variable *t* is time of readings. μ and σ^2 are computed by equations (1) and (2), respectively.

$$\mu = \frac{\sum_{i=1}^{n} T_i}{n} \tag{1}$$

$$\sigma^2 = \frac{\sum_{i=1}^{n}(T_i - \overline{T})^2}{n} \tag{2}$$

where T_i is the time of each reading for an object detected by a reader, n is the number of readings detected by the reader, and $\overline{T}$ is the average time of the readings for the reader. Additionally, we choose $[\mu - 3\sigma, \mu + 3\sigma]$ as the range of independent variable t. Fig. 1(d) depicts the situation based on probabilistic interpolating method.

The processing algorithm of probabilistic interpolating method is formalized as follows:

Algorithm 4. Probabilistic-based interpolating

Input: raw RFID data streams and normal distribution table
Output: the cleaned data streams

1. ***for*** the readings of each object
2. ***if*** missed readings exist according to probability distribution
3. Insert some readings based on normal distribution;
4. ***else***
5. Call a deterministic interpolating method;

4 Experimental Evaluation

In this section, we give experimental evaluations for the proposed methods. We simulate to generate raw data streams on a PC with Pentium (R) dual CPU 1.86 GHz and 2GB of memory. To ensure the experimental results approximate to the reality, we utilize the Netlogo system, which is a well-known simulator, to generate the simulated data. The simulation programs are implemented in C++ running on Linux.

In order to run experiments, we generate three kinds of simulated data sets in an intelligent building scenario. Specifically, three kinds of simulated data sets are described as follows:

Data Set 1: The number of objects is random, and the relationships among objects are random too.

Data Set 2: When tags come into the minor view, due to radio frequency collision, sometime readers miss some readings. Additionally, there are few of containment relationships among tags.

Data Set 3: All the objects will be put into bags or cases and taken by humans. Hence, the readings mainly are produced in the major view.

We compare the proposed methods with some previous methods, which include Static-1, Static-2, Static-5 [5, 6], and SMURF [8]. The main performance metrics are processing time and accuracy. In the following, for convenience, we simply name the time interval-based method, containment relationship-base method, inertia-based method and probabilistic interpolating method as Interval, Relationship, Inertia and

NDPI, respectively. Similarly, we denote the combination of deterministic methods and NDPI as "Deterministic+ NDPI".

Fig. 2 and Fig. 3 show the results of the comparison of processing time over different methods on Data Set 1. We can see from Fig. 2 that processing time grows linearly with the increasing of number of objects, and Inertia has the least processing time consumption while SMURF has the most processing time consumption among these methods. This is because SMURF must compute more parameters for the adaptive changing of time windows. Further, we study the impact of miss ratio of data on processing time. As shown in Fig. 3, Inertia and Relationship have a distinct advantage over the other methods in processing time.

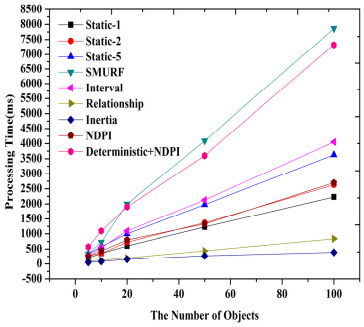


Fig. 2. Comparison of processing time

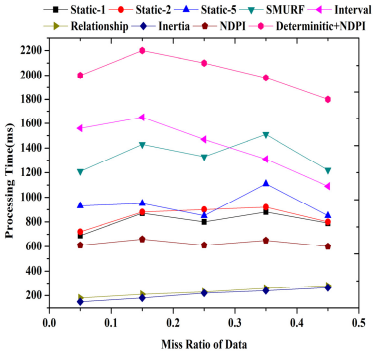


Fig. 3. Processing time vs. miss ratio of data

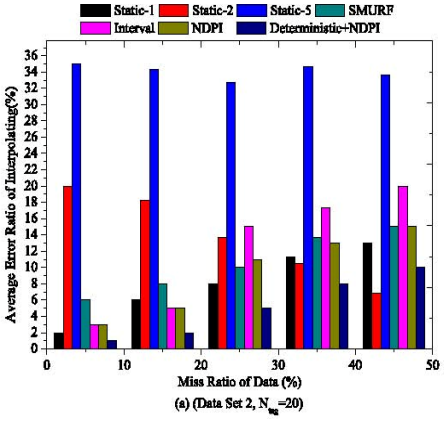


Fig. 4. Accuracy comparison over Data Set 2

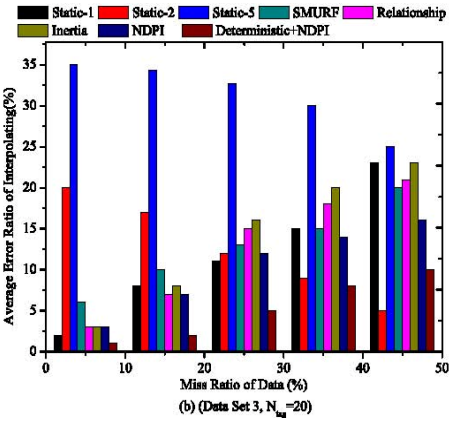


Fig. 5. Accuracy comparison over Data Set 3

We also assess the accuracy of different methods. Fig. 4 and Fig. 5 show the results of the accuracy comparison of different methods. In Fig. 4, the experimental data set (Data Set 2) has large time interval, and the number of tags (N_{tag}) is set to 20. We can

see from Fig. 4 that Interval performs better than Static-*i* methods because there is large time interval between readings on the simulation data set. In Fig. 5, the experimental data set (Data Set 3) has high intensity of inertia and a great numbers of containment relationships. We can learn from Fig. 5 that Inertia and Relationship work well than static-*i* methods in most cases. The reason is there are high intensity of inertia and a great numbers of containment relationships on the simulation data set which is in favor of Inertia and Relationship. The experimental results also demonstrate that small static smooth window shows better behave than the large one. In addition, NDPI works as well as SMURF in most cases.

5 Conclusion

RFID technologies are used in many applications for data collection. However, raw RFID readings are usually of low quality due to frequent occurrences of missed readings and unreliable readings. In this paper, we explore efficient methods for interpolating missed readings. We propose a novel probabilistic interpolating method and three novel deterministic interpolating methods. We conduct extensive experiments, and the experimental results demonstrate the feasibility and effectiveness of our methods.

Acknowledgment. This work is supported by the Natural Science Foundation of China under Grant No. 61170174, 61170027 and the Natural Science Foundation of Tianjin under Grant No. 11JCYBJC26700.

References

1. Chaves, L.W.F., Buchmann, E., Böhm, K.: Finding Misplaced Items in Retail by Clustering RFID Data. In: Proc. of the 13th International Conference on Extending Database Technology. ACM Press (2010)
2. Floerkemeier, C., Lampe, M.: Issues with RFID usage in ubiquitous computing applications. In: Ferscha, A., Mattern, F. (eds.) PERVASIVE 2004. LNCS, vol. 3001, pp. 188–193. Springer, Heidelberg (2004)
3. Rahm, E., Hong, H.: Data cleaning: Problems and current approaches. IEEE Data Engineering Bulletin 23(4), 3–13 (2000)
4. Sarndal, C.E., Swensson, B., Wretman, J.: Model assisted survey sampling. Springer (2003)
5. Franklin, M.J., Jeffery, S.R., Krishnamurthy, S.: Design Considerations for High Fan-in Systems: The HiFi Approach. In: Proc. of the 2nd Biennial Conference on Innovative Data Systems Research, pp. 290–304 (2005)
6. Jeffery, S.R., Alonso, G., Franklin, M.J., Wei, H., Widom, J.: Progressive skyline computation in database systems. In: Proc. of the 22nd International Conference on Data Engineering (2006)
7. Jeffery, S.R., Alonso, G., Franklin, M.J., Hong, W., Widom, J.: Declarative Support for Sensor Data Cleaning. In: Fishkin, K.P., Schiele, B., Nixon, P., Quigley, A. (eds.) PERVASIVE 2006. LNCS, vol. 3968, pp. 83–100. Springer, Heidelberg (2006)

8. Jeffery, S.R., Garofalakis, M., Franklin, M.J.: Adaptive Cleaning for RFID Data Streams. In: Proc. of the 32nd International Conference on Very Large Data Bases, pp. 163–174 (2006)
9. Kanagal, B., Deshpande, A.: Online Filtering, Smoothing and Probabilistic Modeling of Streaming Data. In: Proc. of the 5th ACM International Workshop on Data Engineering for Wireless and Mobile Access, pp. 43–50 (2006)
10. Khoussainova, N., Balazinska, M., Suciu, D.: Towards Correcting Input Data Errors Probabilistically Using Integrity Constraints. In: Proc. of the 24th International Conference on Data Engineering, pp. 1160–1169 (2008)
11. Rao, J., Doraiswamy, S., Thakkar, H., Colby, L.S.: A Deferred Cleansing Method for RFID Data Analytics. In: Proc. of the 32nd International Conference on Very Large Data Bases, pp. 175–186 (2006)
12. Chen, H., Ku, W.S., Wang, H., Sun, M.T.: Leveraging Spatio-Temporal Redundancy for RFID Data Cleansing. In: Proc. of the ACM International Conference on Management of Data, pp. 51–62 (2010)
13. Jiang, T., Xiao, Y., Wang, X., Li, Y.: Leveraging Communication Information among Readers for RFID Data Cleaning. In: Wang, H., Li, S., Oyama, S., Hu, X., Qian, T. (eds.) WAIM 2011. LNCS, vol. 6897, pp. 201–213. Springer, Heidelberg (2011)

A New Indexing Technique for Supporting By-attribute Membership Query of Multidimensional Data

Zhu Wang[1], Tiejian Luo[1], Guandong Xu[2], and Xiang Wang[2]

[1] University of Chinese Academy of Sciences (UCAS),
19(A) Yuquan Rd., Shijingshan District, Beijing 100049, China
[2] University of Technology, Sydney,
P.O. Box 123, Broadway, NSW 2007, Australia
{wangzhu09,wangxiang11}@mails.ucas.ac.cn, tjluo@ucas.ac.cn,
Guandong.Xu@uts.edu.au

Abstract. Multidimensional Data indexing and lookup has been widely used in online data-intensive applications involving in data with multiple attributes. However, there remains a long way to go for the high performance multi-attribute data representation and lookup: the performance of index drops down with the increase of dimensions. In this paper, we present a novel data structure called Bloom Filter Matrix (BFM) to support multidimensional data indexing and by-attribute search. The proposed matrix is based on the Cartesian product of different bloom filters, each representing one attribute of the original data. The structure and parameter of each bloom filter is designed to fit the actual data characteristic and system demand, enabling fast object indexing and lookup, especially by-attribute search of multidimensional data. Experiments show that Bloom Filter Matrix is a fast and accurate data structure for multi-attribute data indexing and by-attribute search with high-correlated queries.

Keywords: Bloom Filter Matrix, Multidimensional data indexing, By-attribute search.

1 Introduction

The rapid growth of Internet in recent years leads to the incredibly fast accumulation of data. With the growth of service type and the more detailed description of objects, many network applications are involved in data with multiple attributes, such as sensor networks [1], [2], text database [3], data stream mining [4], semantic data warehouse [5], Internet video storage [6] and RDF framework [7]. Specially, metadata management [8] in large-scale systems also emphasizes on the integration of data attributes. Despite of the advances in network applications, multi-attribute storage poses a new challenge for large-scale data indexing and search. Unlike one-dimensional data, corpus with multidimensional attributes take up much more space for indexing and the system performance drops down.

Multidimensional data storage enables network systems to record abundant information and offers rich and colourful services to users. Despite of the advances in

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 266–277, 2013.

network applications, multi-attribute storage poses a new challenge for large-scale data indexing and search. Unlike one-dimensional data, corpus with multidimensional attributes take up much more space needed for indexing and the space usage grows rapidly with the increase of dimension.

Besides the space consumption of indexing, most applications based on multidimensional data structure request the support of flexible query mechanism, for not only exact membership queries in which users submit all attributes of a item and look for the object wanted, but also by-attribute membership queries which only contain a part of all attributes of an item. For example, there are many people in a company. Every person has three attributes: post, nationality and age. An exact search contains all the attributes of an item like "Is there a twenty-years-old American accountant in the company?" On the contrary, a by-attribute search can just contain one or a few of all the attributes like this: "Is there an American accountant in the company?" Or simply "Is there an American in the company?" In an exact search of multidimensional data, in order to get the right answer, users must give all the attributes in a query. That increases the difficulties for obtaining objects and restricts the number of search types because in many scenarios, users cannot provide all the attribute information. On the other hand, by-attribute search offers the flexibility for users to obtain the useful information and supports many data operations. Since exact search queries can provide sufficient information for an item, the lookup procedure is easy to implement. On the contrary, the by-attribute search adapts flexible query patterns and therefore complicates the indexing and lookup algorithm.

An efficient way for storing multi-attribute data is to put the same attribute of all data into one container. In the example above, we can establish a table for storing all ages of the staffs in the company. The table is then called "Age Table" and contains ages, e.g. {18, 19, 23, 58, ...}. Similarly we can establish the "Nationality Table" and "Post Table". When a query "Is there a twenty-years-old American accountant in the company?" comes, we can search the "Age Table" for "20", the "Nationality Table" for "America" and "Post Table" for "Accountant". If all the three answers are true, we can judge that the person does work in the company. The method works well when all attribute value are sparsely distributed, i.e. each value is quite distinguishable from another. In case of dense-valued datasets, however, since data attribute values alike, only with one or a few attributes different from another, the lookup method may lose its effectiveness. For example, the corpus is not a company but an American campus where many students age from 18 to 22 and the query remains the same: "Is there a twenty-years-old American accountant here?" Even though all "Age", "Nationality" and "Post" table answer yes, we are not confident enough to say the answer is yes because there are many Americans whose age are twenty in an American campus. That is to say, the more densely the corpus value, the less distinguishable single attribute is, and the less effective the lookup algorithm may be. The densely-valued dataset complicates the multidimensional indexing and lookup.

Bloom filter (BF) [9] is a data structure for representing a set of objects and for supporting membership queries at a cost of a low rate of false positive probability. The simple mathematical format offers a high item indexing and lookup performance and requires a small storage resource. Actually, many online systems, which

emphasize fast item retrieval and can tolerate a small false rate, are now using bloom filters as their key object management component. In multidimensional search, however, most bloom filter variations put all values of the same attribute in one filter and thus suffer from the decrease of accuracy in densely-valued corpus. How to design a fast, accurate and space-efficient algorithm to support by-attribute query in densely-distributed dataset is emerging as a great challenge in multidimensional data indexing and lookup.

In this paper we propose a novel data structure called Bloom Filter Matrix (BFM) to index multidimensional data and devise corresponding lookup algorithms to support exact and by-attribute query. The matrix is based on the Cartesian product of different bloom filters, each representing one attribute of the original data. In that way, the relationship of attributes in the same object is maintained properly. The structure and parameter of each bloom filter is designed to fit the actual data characteristic and system demand, enabling fast object indexing and lookup, especially by-attribute search of multidimensional data.

The rest of the paper is organized as follows: In section two we present the related work of our research. Section three describes our BFM data structure. Section four uses experiments to examine the performance of the proposed algorithm. Finally, section five concludes the paper.

2 Related Work

There have been quite a few previous attempts to apply bloom filters in multidimensional data indexing. Standard bloom filter (SBF) [9] offers an efficient way to represent multidimensional data. It uses a bit vector as an index of items of a set S. When an item is inserted, the algorithm uses a group of hash functions to map the item onto several locations in the bit vector and set the corresponding bit to one. When a membership query is submitted, the algorithm uses the same hash functions to calculate the hash positions of the queried item and check if all the corresponding bits are one. If the answer is yes, bloom filter concludes that the queried item belongs to the set. Otherwise it reports that the query does not belong to the set. It needs to be mentioned that for each item belonging to S, since all the bits of the hash locations are already set to 1, the lookup procedure for the very item will definitely have the positive answer. However, there is a probability that items don't belong to the set be judged as inside S by bloom filter because its hash locations might have been set to 1 by some other items' hashing. That is to say, bloom filter has a false positive rate. Research [10] shows that the false positive rate can be represented as follows:

$$f_{FP} = \left(1 - e^{-\frac{kn}{m}}\right)^k$$

Here m is the bit vector length. n is the item number. k is the hash number and f_{SBF} is false positive rate of SBF. Study [10] also shows that f_{SBF} reaches the minimal value when k values

$$k = \frac{m}{n} \cdot ln2$$

k_{opt_SBF} is the optimal hash number. The time complexity is O(k).

In multidimensional indexing, the standard bloom filter first joins the attributes of an item and then operates with the joint result. In that way, SBF can easily avoid the false rate caused by the combination of attributes. However, the method requires all attributes exist in a query and therefore is not capable of handling by-attribute search.

Guo et al. propose an algorithm called multidimensional dynamic bloom filter (MDDBF) to support data indexing [11]. The algorithm builds one attribute bloom filter as the index for each dimension of the multidimensional set. In item insertion it simply inserts each of dimensional attribute into the corresponding bloom filter. When a query is submitted, the algorithm checks whether its all attributes exist in the corresponding bloom filter. If all the answers are yes, then it reports a positive response. It can be seen that while the method is capable of handling exact and by-attribute queries, it introduces a new type of false positive occurrence: the combination false positive instance. For example, if (a,b) and (x,y) are in the set S, then the query (a,y) will be judged as inside S because both attributes of the query exists in each dimensional bloom filter. Storing the item's attributes of different dimensions separately splits the entire item and the relationship between the attributes of one item is lost. In the example, one cannot tell that attribute a and b belong to one item due to the separate storage.

Multidimensional bloom filter [12] is first invented to store the point-to information in program compiling. The point-to information <p,c,x> stands for pointer, context and pointee, respectively. The algorithm stores the information in a data structure mb[P][C][D][B]. The algorithm uses the one-to-one hash M_p and M_c to map p and c to integers ranging from [0,P-1] and [0,C-1] respectively. The last dimension is a standard bloom filter. A family of D hash functions $\{H_i\}_{i=1...D}$ map the pointee x to D hash locations. Since the first two dimensions of MDBF have a limited size, to avoid mapping collision, the data structure can only record the limited number of records, with the pointer number smaller than P and context value smaller than C. That satisfies the need of point-to analysis well but is not tolerable in some other multidimensional applications.

3 BFM Design

In the section above we analyse the characteristics of multidimensional data indexing techniques. Now we present our new data structure BFM and the indexing algorithm.

3.1 BFM Structure

In BFM, we use d bloom filters to represent d attributes of a multidimensional dataset. Unlike the MDDBF which puts the bloom filters in a parallel group, BFM uses a matrix to present the Cartesian product of those bloom filter vectors. To support by-attribute query, the data structure adds an additional flag position immediately

behind each bloom filter. The structure of BFM is given in Fig.1. (Here we take d=2 as an example). The BFM is a two dimensional matrix, each dimension is a bloom filter that stores one attribute information of items in the dataset (see the gray and white part of Fig.1.). The pink part is flag bits, which is used to support by-attribute query.

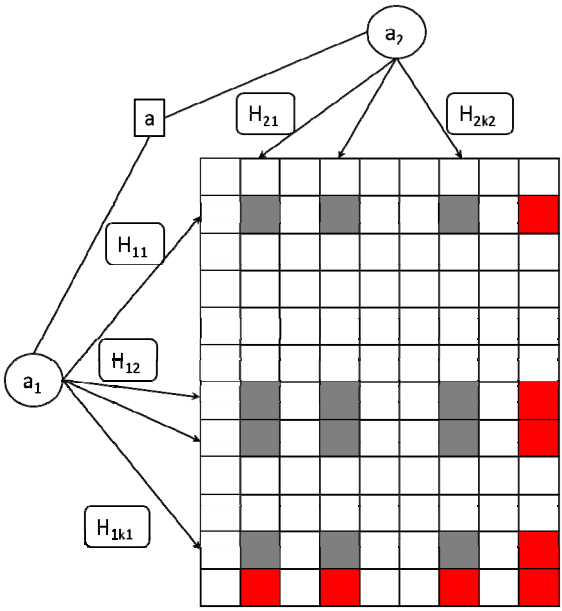


Fig. 1. BFM structure and item insertion

3.2 BFM Implementation

Multi-attribute Indexing and Lookup: A d dimensional item can be presented as $a(a_1,a_2,\ldots,a_d)$. In order to store the multiple attributes of multidimensional data efficiently, we need d bloom filters to store d attributes. Each bloom filter i has a vector length V_i and hash family $\{H_{iki}\}_{1\leq i\leq d}$. To further support the exact membership query and reduce the combine false positive rate, we make BFM the Cartesian product of the bloom filters.

$$BFM = BF_1 \times BF_2 \times \ldots \times BF_d$$

BFM is a d-dimensional matrix and each dimension has a size V_i+1 (later we will explain the reason why the size is V_i+1 not V_i). Hash functions $\{H_{ij}\}_{1\leq i\leq d,1\leq j\leq ki}$, ranging from 0 to V_i-1, are used to map each attribute value onto k_i locations of bloom filter i. When an item a is inserted, the hash function families first compute the hash locations of each item

$$H_i(a_i) = \{h_{ij} | h_{ij} = H_{ij}(a_{ij}), 1 \le i \le d, 1 \le j \le k_i\}$$

We then set the Cartesian product of the hash locations in BFM to one.

$$BFM[P_1][P_2]\cdots[P_d] - 1, P_i \in H_i(a_i), 1 \le i \le d$$

The insertion is demonstrated in Fig.1. Item a has two dimensional attributes a_1 and a_2. Attribute bloom filter one has four hash functions and maps a_1 onto {1,6,7,10}. Attribute bloom filter two has three hash functions and maps a_2 onto {1,3,6}. So the Cartesian product of the two hash location sets: all twelve grey points in BFM are set to 1.

In a lookup procedure of a query $q(Q_1=q_1,Q_2=q_2)$, We use the same hash function families to compute the hash locations

$$H_i(q_i) - \{h_{ij} | h_{ij} - H_{ij}(q_i), 1 \le i \le d, 1 \le j \le k_i\}$$

We find the Cartesian product of H_i's

$$\{(P_1, P_2, \cdots, P_d) | P_i \in H_i(q_i), 1 \le i \le d\}$$

Then we check whether all the corresponding BFM values equal to 1. If yes, we conclude that the element belongs to the set. Otherwise, we are sure that the element does not belong to the set.

In the BFM design we can see that the construction of the data structure reflects the nature of multidimensional dataset: Cartesian product of single attributes. The dimensional information carrier, attribute bloom filters make Cartesian product to represent the corresponding multidimensional dataset. A query receives a positive answer only when all attributes hash locations' Cartesian product position report yes. In that way, the internal association between an item's attributes are preserved and the combined false positive occurrence is avoided. Actually, BFM can support not only high accuracy exact membership query but also by-attribute query.

By-attribute Query Support. In by-attribute query, we cannot obtain all the attribution information of the queried item. Let's look at a two dimensional example. In a two dimensional dataset $S=(D_1, D_2)$, we want to find out whether there is an item whose first is a in the set. So the query is $q(D_2=q_2)$. We use hash family $\{H_{2j}\}_{j=1\ldots k2}$ to compute the k_2 locations in the second dimension (column) of BFM. If there is an item whose second attribute value is "q_2", the corresponding columns will all have elements equal to one. So we have to check each column whose column index belongs to $\{H_{2i}(\text{“}q_2\text{”})\}_{i=1\ldots k2}$ to find out if there is a non-zero element in each column.

In BFM, we add an extra bit (flag bit) at the end of each dimension vector to represent the number of ones in that dimension (see pink positions in Fig.1., that is why the length of the ith dimension filter is V_i+1). The flag bit is set to 1 as long as one of the elements in that dimension is 1 (see red positions in Fig.1. They are set to 1 because there is at least one of the positions in the same dimension being 1.)

$$\forall J_i \in \{1,2,\cdots,d\} i = 1,2,\cdots,L$$
$$BFM[V_1][V_2]\cdots[P_{J_1}]\cdots[P_{J_2}]\cdots[P_{J_L}]\cdots[V_d] - true$$
$$iff \exists J_F \in \{1,2,\cdots,d\}$$
$$s.t. BFM[P_1][P_2]\cdots[P_{J_1}]\cdots[P_{J_2}]\cdots[P_{J_F}]\cdots[P_{J_L}]\cdots[V_d] = true$$

In actual insertion progress, once we set an element in BFM to 1, we set the corresponding flag bits to 1 as well (see Fig.1.).

In by-attribute membership query, when a query $q(Q_{J1}=q_{J1}, Q_{J2}=q_{J2}, \ldots, Q_{JE}=q_{JE})$ is submitted (For easier expression, we may assume that $J_1=1$, $J_2=2$, …, $J_E=E$ and $q(Q_1=q_1, Q_2=q_2, \ldots, Q_E=q_E)$), we simply calculate the hash positions of each element $\{H_{ij}\}_{1\le i\le E, 1\le j\le ki}$ and check if all the corresponding flag bits are true.

$$q \in S$$
$$iff \text{ every } BFM[H_{1J_1}(q_1)][H_{2J_2}(q_2)]\cdot[H_{EJ_E}(q_E)][V_{E+1}][V_{E+2}]\cdots[V_d] = 1$$
$$1 \le J_i \le K_i, 1 \le i \le E$$

BFM can achieve a much smaller false positive rate with high correlated queries because our solution excludes combination false positive occurrences.

BFM allows designers to choose different algorithms for each dimension according to the usage environment. For each dimension, we can easily replace the standard bloom filter with other indexing algorithms. For instance, if one attribute of the dataset has a compact static value range (e.g. gender), we can use a perfect hashing [13] to replace the bloom filter in that dimension. If we can obtain the prior knowledge of access frequency of one dimension's elements, we can use the weighted bloom filter [14] instead of the standard bloom filter. The structure of BFM allows independent combination of different indexing technique for each attribute dimension.

4 Experimental Evaluation

We do experiments to test the performance of the BFM algorithm. The experiments are mainly conducted from four aspects: exact membership query, by-attribute membership query, time consumption and hash number effect.

4.1 Exact Membership Query

In this section we aim to test the accuracy of BFM exact membership query. In the experiment, the BFM is a two dimensional matrix. Each dimension represents an attribute bloom filter. Both bloom filters are 2^{10} bits long and have three independent hash functions. The total hash function number is six. The BFM is previously loaded with a storage corpus of 10,000 items.

We use approximately one hundred million queries to test the BFM accuracy. All query attributes are fetched from a dynamic dataset. In order to reflect the density of attribute value, we define a new parameter: the overlap rate α. α is the probability that an attribute value of a query has a same copy in the stored corpus. So $\alpha=0$ means that

all attributes of queried items are different from any one of the attributes in the stored corpus. On the other hand, α=1 means all attributes of a queried item have a copy in the corpus. The larger the α is, the more correlated the queries are. Note that the both attributes of a query corresponding to a copy in the stored corpus does not necessarily mean that the query belongs to the corpus because of the many combinations of both attributes. The two attributes may belong to different items in the corpus. Actually in our experiment, we exclude the counting of queries that belong to the corpus: no query can find an exact match in the dataset.

In the experiment, we use three other bloom filter variations as the baselines of the test: SBF, MDDBF and MDBF. In order to make a fair comparison, we set their size and the hash number to be same with the BFM's. The false positive probability of the four algorithms is shown in Fig.2.

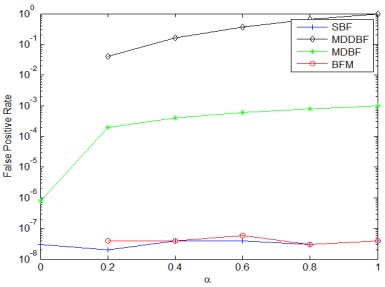


Fig. 2. False positive rate of exact membership search

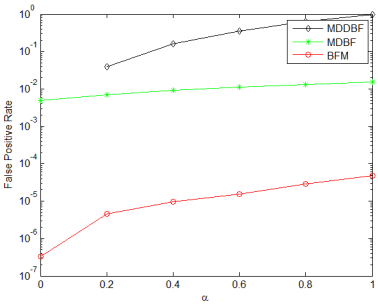


Fig. 3. False positive rate of by-attribute membership search

In the figure we can see that the accuracy of the SBF and the BFM is very close to each other. The two algorithms are the best among the four. (Although SBF doesn't support by-attribute query.) Their false positive rate remains stable as the overlap rate rises. It needs to mention that the false positive rate of BFM and SBF is much lower than those of other two algorithms, even five orders of magnitude lower when α=1. That means the BFM handles the high correlated queries very well. MDBF false positive is higher than BFM because it uses one hash function for a certain dimension which increases the hash false rate. MDDBF performs worst with high correlated queries (with false rate 100% when α=1) because it cannot tell which attribute belongs to which item. As long as both attributes exist in the corpus, the query is judged as inside S. That combination false positive rate is the reason for high false rate with large α.

4.2 By-attribute Membership Query

This section reports the by-attribute query performance of BFM. In the experiment, we use a three dimensional BFM matrix to index 3-D items of a corpus. Each dimension represents an attribute bloom filter. All the three bloom filters have 2^8 bits.

We use three hash functions for each bloom filter. The BFM is loaded with one thousand items. At the query side, we have three million queries. Each query has two dimensions. Since SBF does not support by-attribute query, we use MDBF and MDDBF as the baseline of our BFM algorithm. To make a fair comparison, the two algorithms have the same size and hash function number as the BFM.

The experimental result is given in Fig.3. The result shows that the three algorithms' false positive rates grow higher with the increase of overlap rate. Among them, the BFM has the lowest false rate. Actually, the false positive difference can achieve up to four orders of magnitude. MDBF is ranked the second due to the lack of hash functions in two dimensions. MDDBF performs worst because it cannot get rid of combined false positives.

4.3 Time Consumption

We keep the time for the above exact search and by-attribute search. The time consumption of the exact search and the by-attribute search is given in Table 1 and Table 2.

Table 1. Time per Query (ms) for Exact Search

	α=0	α=0.2	α=0.4	α=0.6	α=0.8	α=1
SBF	0.0501	0.0503	0.0502	0.0505	0.0503	0.0503
MDDBF	0.0445	0.0531	0.0615	0.0696	0.0788	0.0882
MDBF	0.0838	0.0832	0.0829	0.0804	0.0821	0.0774
BFM	0.0841	0.0815	0.0776	0.0709	0.0627	0.0566

Table 2. Time per Query (ms) for by-attribute Search

	α=0	α=0.2	α=0.4	α=0.6	α=0.8	α=1
MDDBF	0.0411	0.0491	0.0571	0.0657	0.0735	0.082
MDBF	0.0752	0.0746	0.0752	0.0752	0.0748	0.0753
BFM	0.0756	0.0745	0.0753	0.0753	0.0747	0.0755

From table 2 and table 3 we can see the four indexing algorithms have the similar time consumption. That is because the majority of computation time is used for hashing. Since the algorithms are configured to have the same hash function number, the similar time consumption is expectable.

We do another experiment to show the relationship between the number of items stored and query time. In the experiment, we use a three dimensional BFM matrix to index 3-D items of a corpus. Each dimension represents an attribute bloom filter. All the three bloom filters have 256 bits. We use one hash function for each bloom filter, three functions in total. Item loaded in the BFM varies from one thousand to one million. We perform exact search and by-attribute search in the BFM. The result is given in Table 3.

Here ES is short for exact search, BS is short for by-attribute search, TPQ is short for time per query and FP is short for false positive rate.

In Table 3 we can see that the time used for searching for one item remain almost the same even when the total number of stored items increases. That is a good

reflection of the constant time complexity for bloom filters. Given the fixed hash numbers $\{h_i\}_{i=1...d}$, the time complexity of our algorithm is O(C), where C is the sum of all attribute bloom filter hash numbers.

Table 3. Time Per Query with Different Corpus Size

Corpus	ESTPQ(ms)	ES FP	BSTPQ(ms)	BS FP
1000	0.070949	0.000048	0.046238	0.001573
8000	0.070715	0.000432	0.04602	0.005635
27000	0.071074	0.001386	0.046134	0.012313
64000	0.071027	0.002058	0.046056	0.018672
125000	0.071043	0.0045	0.046127	0.030017
216000	0.07098	0.009072	0.04629	0.041477
343000	0.070965	0.012512	0.046186	0.055063
512000	0.071214	0.013824	0.046098	0.070725
729000	0.070965	0.02244	0.045942	0.085234
1000000	0.071214	0.029295	0.046056	0.099831

4.4 Hash Number

We use a two dimensional BFM to demonstrate the hash number effect on BFM. The single bloom filter size is 1024. Hash numbers of the two bloom filters are the same. The stored corpus and query set are the same as in section 4.1.

We choose the hash number of the single bloom filter from 1 to 20. The false positive rate is given in the Fig.4.

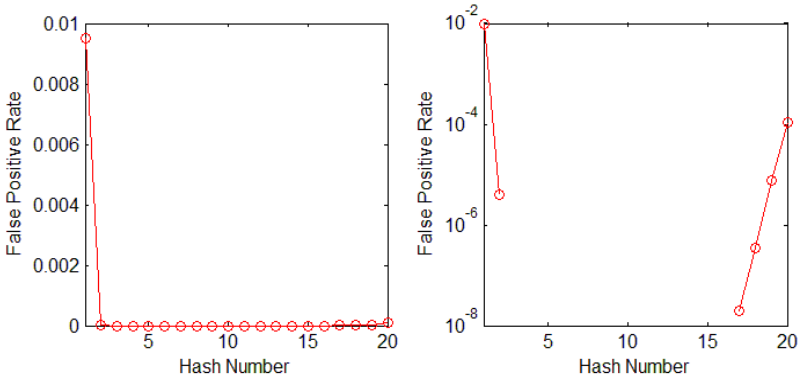


Fig. 4. Hash number effect, linear and log scale

The left figure is in a linear scale. To be clearer, we plot a figure in a log-log scale on the right. Here we can see that the BFM's hash effect follows the same tendency of that of bloom filter. There exists a long region whose false positive rate is very low. Therefore, this brings in the great convenience in choosing hash function number, and increases the usability of BFM.

4.5 Space Usage

In terms of space usage, we use the three dimensional BFM as the index of 3-D corpus. We change the corpus size and BFM size and calculate the false positive rate of exact membership search and by-attribute membership search. For each corpus size, we use one million 3-D queries for exact search and three million 2-D queries for by-attribute search. The false positive rate and space used per item (SUPI) is given in Table 4. Here FP is the short for false positive rate, ES is for exact membership false, BS is for by-attribute membership search and SUPI is short for space used per item.

Table 4. Space Usage of BFM

Corpus	BFM Size	SUPI	FP for ES	FP for BS
512	4913	About 8	0.08151	0.1877
4096	35937		0.078584	0.1603
32768	274625		0.070875	0.1583
262144	2146689		0.061705	0.1582
2097152	16974593		0.05852	0.1435
16777216	135005697		0.05712	0.1443
134217728	1076890625		0.055944	0.152
64	4913	About 64	0.0078	0.0345
512	35937		0.00442	0.0250
4096	274625		0.0069	0.0279
32768	2146689		0.004968	0.0261
262144	16974593		0.002535	0.0250
2097152	135005697		0.002145	0.0259
16777216	1076890625		0.004998	0.0269

When SUPI=8, the average false positive rates of exact search and by-attribute search are 0.0663 and 0.1578 respectively. When SUPI=64, the average false positive rates of exact search and by-attribute search are 0.0048 and 0.0273 respectively.

Seen from the table, the highest performance of BFM is achieved by using 64 bits to represent a three dimensional item. In a table based index, that space of 64 bits is able to accommodate just one of the three attribute of the item. The space efficiency of BFM is even much higher with SUPI=8 and the average false positive rate reaches 0.0663.

5 Conclusions

In this paper we have defined two kinds of membership query for multidimensional data: exact search and by-attribute search. The exact membership query retrieves all attributes of queried item and returns the membership information of the very item mentioned in the query. The by-attribute membership query only provides some of the attribute values and asks if there is an item that satisfies the queried needs. We have shown that the by-attribute query is an emerging challenge for multidimensional indexing, especially with high correlated queries.

We have analyzed three existing indexing strategies of multidimensional data. SBF has very high performance but do not support by-attribute query. MDBF and MDDBF can support by-attribute query but do not perform well with high correlated queries

Then we proposed a novel multidimensional indexing technique BFM. The matrix is a Cartesian product of several bloom filters. Each bloom filter represents an attribute of the multidimensional data. We have proved that BFM can support by-attribute membership query and achieves high performance with high correlated queries.

We have carried out experiments to validate the performance of BFM in four aspects: exact query false positive rate, by-attribute query false positive rate, time consumption and hash number effect. The experiments have shown that BFM is a fast and accurate data structure for multi-attribute data indexing and by-attribute search with high correlated queries.

References

1. Li, X., Kim, Y.J., Govindan, R.: Multi-dimensional Range Queries in Sensor Networks. In: SenSys 2003, pp. 63–75. ACM, New York (2003)
2. Liu, B., Lee, W.-C., Lee, D.L.: Distributed Caching of Multi-dimensional Data in Mobile Environments. In: MDM 2005, pp. 229–233. ACM, New York (2005)
3. Lin, C.X., Ding, B., Han, J., Zhu, F., Zhao, B.: Text Cube: Computing IR Measures for Multidimensional Text Database Analysis. In: ICDM 2008, pp. 905–910. ACM, New York (2008)
4. Jiang, N., Gruenwald, L.: Research Issues in Data Stream Association Rule Mining. ACM SIGMOD Record 35, 14–19 (2006)
5. Nebot, V., Berlanga, R., Pérez, J.M., Aramburu, M.J., Pedersen, T.B.: Multidimensional Integrated Ontologies: A Framework for Designing Semantic Data Warehouses. Journal on Data Semantics 13, 1–36 (2009)
6. Wang, Z., Luo, T.: Intelligent Video Content Routing in a Direct Access Network. In: SWS 2011, pp. 147–152. IEEE Press (2011)
7. Berners-Lee, T., Connolly, D., Kagal, L., Scharf, Y., Hendler, J.: N3Logic: A Logical Framework for the World Wide Web. Theory and Practice of Logic Programming 8, 249–269 (2008)
8. HüNer, K.M., Otto, B., ÖSterle, H.: Collaborative Management of Business Metadata. International Journal of Information Management: The Journal for Information Professionals 31, 366–373 (2011)
9. Bloom, B.: Space/time Trade-offs in Hash Coding with Allowable Errors. Communications of the ACM (CACM) 13, 422–426 (1970)
10. Mullin, J.: A Second Look at Bloom Filters. Communications of the ACM 26, 570–571 (1983)
11. Guo, D., Chen, H., Luo, X.: Theory and Network Applications of Dynamic Bloom Filters. In: INFOCOM 2006, pp. 1–12. IEEE Press (2006)
12. Nasre, R., Rajan, K., Govindarajan, R., Khedker, U.P.: Scalable Context-Sensitive Points-to Analysis Using Multi-dimensional Bloom Filters. In: Hu, Z. (ed.) APLAS 2009. LNCS, vol. 5904, pp. 47–62. Springer, Heidelberg (2009)
13. Belazzougui, D., Boldi, P., Pagh, R., Vigna, S.: Theory and Practice of Monotone Minimal Perfect Hashing. Journal of Experimental Algorithmics, Article No. 3.2 (2011)
14. Bruck, J., Gao, J., Jiang, A.: Weighted Bloom Filter. In: ISIT 2006, pp. 2304–2308. IEEE Press (2006)

CorePeer: A P2P Mechanism for Hybrid CDN-P2P Architecture

Huafeng Deng[1, 2] and Jun Xu[1]

[1] South China Normal University, China
xuj@scnu.edu.cn
[2] Jiangxi Normal University, China
dhfpap@sohu.com

Abstract. Hybrid CDN-P2P live streaming systems are designed and deployed to achieve the scalability of P2P networks and the desired low delay and high throughput of CDNs. However, the P2P mechanism used in pure peer-to-peer systems is not suitable to directly be applied in the hybrid CDN-P2P systems where edge server with strong service capacity is deployed. So, we customize a P2P mechanism, called PeerCore, for hybrid CDN-P2P architecture. PeerCore achieves dynamic resource scaling while guaranteeing quality-of-service by controlling the number of end users directly served by the edge server. In the mean time, we define a superiority index to reflect the stability and upload bandwidth capacity of a node. The more stable end hosts and end hosts with higher upload bandwidth have larger values of superiority index and gain the privilege to receive data directly from edge server. End hosts with larger value of superior index are apt to be promoted closer to edge server and consist of the delivery backbone for the live streaming system. Simulations results demonstrate the superior efficiency and robustness of PeerCore.

Keywords: Content delivery networks, peer-to-peer, live streaming.

1 Introduction

Content Delivery Networks (CDNs) and Peer-to-Peer (P2P) systems are two alternative and competing technologies for delivering Internet streaming. Both approaches have their advantages and disadvantages. CDNs provide excellent quality to end users when the workload is within the provisioning limits. However, CDN servers are expensive to deploy and maintain, and consequently incur a cost for media providers and/or clients. On the other hand, P2P technologies solve the scalability issue by leveraging the resources of the participating peers while keeping the server requirements low. Delayed arrivals of data packets, probably caused by network congestions, retransmissions for packet losses, or peer dynamics in P2P overlay, will lead to rebuffering and playback discontinuities on the client side.

It is a nature idea to build a hybrid CDN-P2P architecture that incorporates the best of both technologies and mutually offsets each others' deficiencies. Several companies such as Akamai, CNN and BitTorrent Inc. combined P2P technology with

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 278–286, 2013.

its back-end CDN system to reduce startup delay and improve quality of services [1-4]. Several researchers have in fact hypothesized and analyzed the potential benefits of the approach via simulations and trace-driven analysis. The work presented in [5] is one of the early analyses on hybrid CDN-P2P architectures. Authors perform an in-depth analysis of the system dynamics which involves a transition between a CDN mode to a P2P mode and use simulations to demonstrate that such a hybrid approach is cost-effective. Huang et al. [6] presented the potential savings in using hybrid CDN-P2P systems for Akamai and Limelight. Many others research papers also recommend the use of hybrid systems. Most of these works are based on simulations and traces collected from pure CDNs. A very detailed analysis of a real-world hybrid CDN-P2P architecture is described in [7]. They present the design, implementation and evaluation of LiveSky, a hybrid CDN-P2P live streaming system developed and deployed by China Cache.

Compare to end hosts, each CDN edge server always has high bandwidth and large capacity of storage and are more stable. So, it can serve more "peers". In order to make a tradeoff between the operating cost of the CDN, the perceived user quality and the overall number of end users that can be supported by the system at a given time in a hybrid CDN-P2P environment. The number of end users served directly by the CDN edge server should varied with the total number of end users, the bandwidth capacity of the edge servers, bounds on user quality defined as the delay between the video source. Each edge server in LiveSky operates in four stages. We borrow the idea of dynamic resource scaling from LiveSky and unite the step 2 and step3 of edge server operation in LiveSky into a single step to avoid uneven transition.

The upload bandwidth of the leaf nodes are wasted if the nodes are organized into a tree rooted at the edge server. So, mTreebone which is a hybrid tree/mesh design construct a tree-based backbone by a set of stable nodes and in the meantime an auxiliary mesh overlay to accommodate node dynamics and fully exploit the available bandwidth of overlay nodes [8]. Further study discovers that stable peers and peers with high upload bandwidth contribute more resources to live P2P multimedia streaming systems [9]. Instead of using existing P2P technologies directly in the CDN-P2P hybrid architecture, we integrate several technologies to design a novel P2P mechanism, referred to as CorePeer.

The remainder of this paper is organized as follows: The architecture of mTreebone is described in Section 2. Details about CorePeer construction and evolution are discussed in Sections 3. We evaluate the performance of CorePeer in Section 4. Finally, Section 5 concludes the paper.

2 Overview of the P2P Mechanism in the Hybrid CDN-P2P Architecture

The hybrid CDN-P2P architecture consists of a set of core servers, a set of edge servers and end hosts. Core servers are responsible for managing the CDN, saving the content being distributed and when needed forwarding content to edge servers. We consider a video streaming system using hybrid CDN-P2P architecture as shown in

Fig. 1. We only show one CDN edge server because we focus on the interaction between one edge server and the clients in its service domain. The edge server in its service domain serves two roles: (1) the actual media streaming server for the clients assigned to it and (2) a tracker for the P2P operation to bootstrap new clients with candidate peers. The client is then redirected to this edge server using traditional DNS-based redirection techniques. The clients engage in P2P transfers. Each client registered for the media file also plays two roles: (1) before receiving the streaming service, the client is a requesting peer. (2) After receiving the streaming service, it becomes a supplying peer. In addition, a node may gracefully leave the overlay, or abruptly fail without any notification.

Among all the clients, these client directly served by the edge server are called core peers. Core peers should be the peers which are stable or have high upload bandwidth capacity. We define a superiority index to reflect the stability and bandwidth capacity of a node in Section 3. Peers with higher superiority index are position closer to the edge server and make up of the backbone for each per-edge server domain.

The upload bandwidth of the leaf nodes are wasted if the nodes are organized a tree rooted at the edge server. In order to improve the resilience and efficiency of the, we further organize all the nodes except edge server into a mesh overlay similar to mTreebone [8]. In this auxiliary mesh overlay, each node keeps a partial list of the active overlay nodes and their status. However, a node will not actively schedule to fetch data blocks from neighbors using such data availability information until data outage occurs in the treebone.

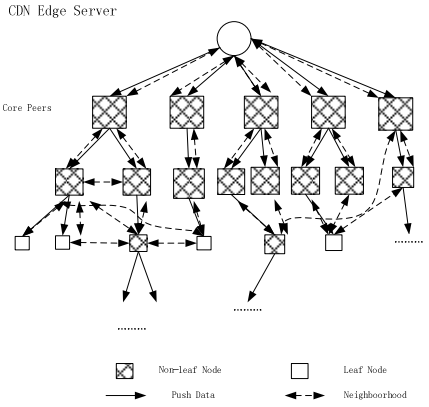


Fig. 1. The hybrid architecture for streaming media distribution

3 CorePeer Construction and Optimization

3.1 Stable End Host

In our designed architecture, the edge server directly serves only a subset of the nodes, in particular, the stable nodes. These stable nodes further provide the streaming

data for the other nodes. It is obvious that even a small set of stable nodes is sufficient to support the entire overlay [7].

The critical question here is thus how to identify stable nodes. Recent studies have found that nodes with a higher age tend to stay longer in overlay multicast systems [7]. We adopt an optimal threshold-based method proposed in [7] to identify stable nodes. According to the method, although the node's duration in the overlay cannot be known before the node actually leaves; the time elapsed since its arrival can be used to predict it. If its age is above a certain threshold, a node can be considered stable.

The age threshold for a node arriving at time t is roughly set to 0.3(L – t), where L is the length of the multicast session. In other words, the age threshold for a node arriving at time t is 30% of the residual session length. A node has chance to be served by the edge server when its age exceeds the corresponding threshold.

3.2 Superiority Index

Let A_i be the age of node *i* in the session, i.e., the time elapsed since its arrival, and UP_i be upload bandwidth of node *i*. We further define a superiority index to reflect the relative potential of contribution to the system of two end users.

$$\mathrm{Super}(u_i, u_j) = w.\frac{A_i - A_j}{(min(A_i - A_j) + 1).\mathrm{avg(A)}} + (1 - w).\frac{UP_i - UP_j}{(min(UP_i - UP_j) + 1).avg(UP)} \quad (1)$$

w is the weight coefficient and can be adjusted according to different scenarios.

Superiority index is proposed to distilling superior peers during streaming, which can potentially stay in the system for a long time and contribute a high level of upload bandwidth. The peers with high value of superiority index should be position closer to the edge server.

3.3 Adaptive Scaling

The working environment of the edge server such as the upload bandwidth capacity of clients, churn rates, the total number of end users changes continuously with the coming and going of the end users. So, we argue that the number of end users served directly by the edge server should vary with the fluctuation of the workload in order to make tradeoffs in a hybrid CDN-P2P environment between the operating cost of the CDN, the perceived user quality, and the overall number of end users that can be supported by the system at a given time.

Suppose the edge server has sufficient capacity to serve only N_c concurrent streaming sessions. Let P_0 be the total number of clients assigned to this edge server that wish to receive this live video stream. Let P_e be the total number of clients receive live video stream directly from this edge server. Let ρ denote the average fraction of full video bitrate that each supplying peer contributes during a session.

We model the node churn using two parameters: the average leave rate σ and the average join rate λ. σ and λ are expressed as a fraction of the total number of current peers. Let K_0 denote the maximum level in the P2P overlay; i.e., a parameter to capture the maximum acceptable source-to-end delay. The service enlargement coefficient can be defined as follows:

$$S = \frac{N_c}{P_0} = \begin{cases} \frac{(1-\rho)(1+\lambda-\sigma)^{K_0}}{(1-\rho^{K_0})(1-\sigma)^{K_0-1}} & \text{if } \rho \neq 1 \\ \frac{(1+\lambda-\sigma)^{K_0}}{K_0(1-\sigma)^{K_0-1}} & \text{if } \rho = 1 \end{cases} \quad (2)$$

As shown in Fig. 2, each edge server can be in three stages (where N_b is simply the total bandwidth capacity of the edge server divided by the media rate).

—Stage 1, $P_0 \leq uN_b$. u is usually set to 0.5 in most scenarios. It is safe and efficient for the edge server to spent fifty percents capacity on sending data to the clients. All clients receive the content directly from edge server. In stage 1, since the number of end user is low, edge server has enough capacity to send data directly to the end users.

—Stage 2, $uN_b \leq P_0 \leq N_b/S$. It is time to exploit the unused capacity of the end users. Whether a new user is served by the edge server or the peers depends on the value of P_e/P_0. If the value of P_e/P_0 is less than S which is defined in equation (2), the new user is assigned to edge server. Otherwise it is redirected to a peer.

—Stage 3, $P_0 \geq N_b/S$

The edge server hits its capacity limit. New clients are redirected to other less loaded edge servers by CDN's GSLB technology.

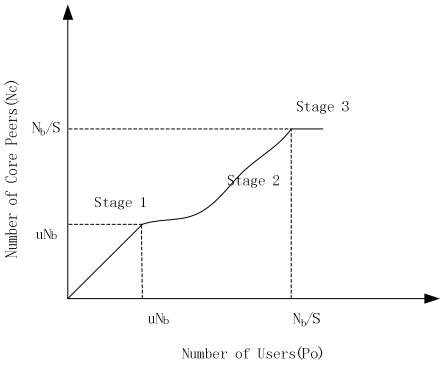


Fig. 2. Different stages in the operation of an edge server

3.4 Bootstrapping and Evolution

Each time a new user *x* enter, the new user become a core peer if the quota of core peers does not run out. Otherwise, the edge server will check its potential to preempt the position of some core peer. The edge server computes the superiority index between the new user and every core peer. Then the largest superiority index among

them is chosen. Suppose the largest superiority index achieve between *x* and core peer y and is greater than a threshold δ, user *x* will then preempt y's position and *y* can simply attach itself to *x*. otherwise, the edge server will assign user *x* to the subtree rooted at a core peer whose total upload bandwidth is the lowest. The core peer in the subtree will find a proper position for *x* in its subtree by computing superiority index in the same way.

If a node is not in the core peers, it will periodically check its own age in the overlay. Once its age exceeds the threshold, 30% of the residual session length, it will promote itself closer to edge server or become a core peer by computing superiority index.

A node may gracefully leave the overlay, or abruptly fail without any notification. Similar methods as proposed in [7] are used here to handling dynamics.

4 Performance Evaluations

In this section, we present extensive simulation results to demonstrate the effectiveness of the hybrid architecture, CorePeer. We simulate a hybrid system with one CDN server and there are 5000 clients registered for a media file. Unless stated otherwise, the simulation parameters are set as follows:

Initially, the CDN server reserves a capacity for serving N_b = 500 simultaneous streaming sessions for the media file to be distributed. We have also implemented an application layer multicast systems, mTreebone [7], for comparison. mTreebone is a novel hybrid tree/mesh design.

In our evaluation and comparison, we use the following three typical metrics, which combined reflect the quality of service experienced by users.

Startup latency, which is the time taken by a node between its requesting to join the session and receiving enough data blocks to start playback;

Transmission delay, which is the time to deliver a data block from the source to the node. We use second as the unit;

Data loss rate, which is defined as the fraction of the data blocks missing their playback deadlines, i.e., either lost during transmission or experienced excessive delays.

We adopt a dynamic scenario for the evaluation, where the overlay nodes arrive at different times and may also leave or fail before the session ends. We use the Pareto distribution to model the durations of the nodes in a session. Unless otherwise specified, the following default parameters are used in our simulation, most of which follow the typical values reported in [7]. The session length L is set to 6000 seconds and each data block is of 1-second video; the maximum end-to-end delay is 1000 ms between two overlay nodes, and the maximum upload bandwidth is uniformly distributed from 4 to 12 times of the bandwidth required for a full streaming. We set the default age threshold to 30% of the residual session length. In general, a larger k means the overlay is of a higher churn rate, i.e., more dynamic. K is set to be 1, a representative value for performance evaluation as in [7]. Estimating of ρ = 1,

λ = 3.2%, and σ = 2.8% are obtained from earlier test deployments. The superiority index threshold δ is set to 0.2.

Fig. 3. shows the CDF of the startup latency. We can see that CorePeer has the lower latency than mTreebone. This is because that mTreebone does not fully explore the capacity of edge server and the peers with high superiority index, although a new node in mTreebone can receive data in the tree structure before establishing the mesh neighborship, and the high-degree-preemption and low-delay-jump minimize the delay of the tree bone. In CorePeer, some core peers or new users can receive data directly from the edge server and the number of core peers are controlled to make tradeoffs between the operating cost of the CDN and the perceived user quality. The peers with high superiority index are more stable or have higher upload bandwidth than the other peers. They consist of the efficient backbone for the other peers. Similar reasons also explain the results of transmission delay in Fig. 4. The data loss rate for the two systems is given by Fig. 5. CorePeer also outperforms mTreebone, validating the advantage of the CorePeer design. Although the data loss rates are generally low in two systems, the backbone based on peers with high capacity and the edge server further reduce the data loss rates in CorePeer.

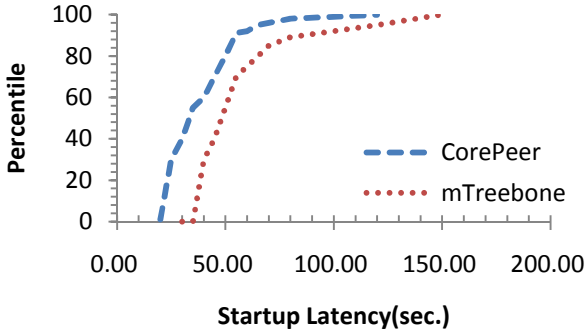


Fig. 3. CDF of startup latency

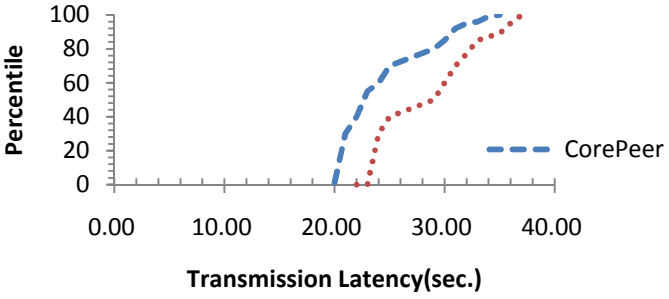


Fig. 4. CDF of transmission latency

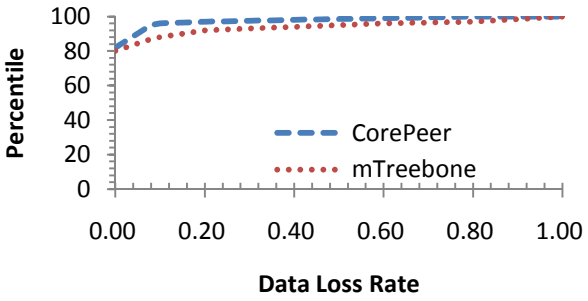


Fig. 5. CDF of data loss rate

5 Conclusions

Instead of directly using existing P2P technologies in the CDN-P2P hybrid architecture, we integrate several technologies to design a novel P2P mechanism. In order to fully explored the opportunity to leverage the capacity of edge server within the CDN-P2P hybrid architecture, the number of peers directly served by the edge server is controlled according to the total number of end users, the bandwidth capacity of the edge servers, user expectation on quality, the upload bandwidth capacity and churn rates of end users. Specially, the edge server prefers to delivers data to core peer which has high upload bandwidth or stability. We define superiority index to measure the potential to contribute for the system. The peers with high value of superiority index are apt to be promoted closer to edge server and consist of the backbone for the streaming system.

We evaluated the performance of CorePeer and compared it with mTreebone. The simulation results demonstrated the superior efficiency and robustness of this hybrid solution. In our future work, we plan to design the mechanism of automatically revising some system parameters such as join rate, leave rate, actual workload state of the edge server to reflect the dynamics of the hybrid CDN-P2P working environment.

References

1. Homer, B.: Akamai using P2P for enhanced video delivery, http://www.onlinevideowatch.com/akamai-using-p2p-for-enhance-video/
2. Octoshape Grid Delivery Enhancement for Adobe Flash Player, http://www.octoshape.com/addin/about.php
3. BitTorrent Content Delivery Service, http://www.bittorrent.com/dna
4. Pando Managed P2P Network p2p, http://www.pandonetworks.com/
5. Xu, D., Kulkarni, S., Rosenberg, C., Chai, H.: Analysis of a CDN–P2P hybrid architecture for cost-effective streaming media distribution. Multimedia Systems 11, 383–399 (2006)

6. Huang, C., Wang, A., Li, J., Ross, K.: Understanding hybrid CDN–P2P: why limelight needs its own red swoosh. In: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 75–80 (2008)
7. Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., Li, B.: Livesky: Enhancing cdn with p2p. ACM Trans. Multimedia Comput. Commun. Appl. 6(16), 1–16 (2010)
8. Wang, F., Xiong, Y., Liu, J.: mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming. IEEE Trans. Parallel and Distributed Systems 21(3), 379–392 (2010)
9. Liu, Z., Wu, C., Li, B., Zhao, S.: Distilling superior peers in large-scale P2P streaming systems. In: INFOCOMM, pp. 82–90 (2009)

Extracting Prevalent Co-location Patterns from Historic Spatial Data

Lizhen Wang, Pingping Wu, Gaofeng Fan, and Yongheng Zhou

Department of Computer Science and Engineering, Dianchi College,
Yunnan University, Kunming 650091, China
lzhwang@ynu.edu.cn

Abstract. A co-location pattern is a set of spatial features whose instances frequently appear in a spatial neighborhood. The data from which the patterns are derived is often historic, yet little or no attention has been paid to the time aspects of this data. In this paper we study the problem of finding prevalent co-location patterns from time constrained spatial data. We define spatial instances with time constraints brought to a net present value, and then define weighted row instances, weighted table instances and a weighted participation index for the spatial co-location patterns. We propose two algorithms to extract prevalent co-locations from spatial data with time constraints, a w-join-based algorithm that can find all prevalent patterns, and a top-*k*-w algorithm to find the top *k* most prevalent co-location patterns. Optimization strategies for the two algorithms are presented. Finally, we show the performance of the proposed algorithms using "real+synthetic" data sets, including the effect of various parameters on the algorithms.

Keywords: Spatial co-location pattern mining, net present value time constraints, weighted participation index, top-*k*.

1 Introduction

The extraction, or mining, of spatial co-location patterns is a rising and promising field in spatial data mining. Such mining discovers correlations and relationships between spatial features. Examples of spatial features include plant species, animal species, business categories, geotechnics, diseases, crime, climate, etc. The data records for these spatial features always includes spatial instances, an occurrence of a spatial feature at a certain location, and often includes a time attribute, because there may be different spatial instances in the same location but at different times. However, previous research on the mining of co-location patterns has not considered the time attribute of spatial instances.

In this paper, we consider the time dimension as a constraint in mining spatial co-location patterns. We define the familiar concepts of a weighted participation index and of the prevalence of co-locations, but both now with time constraints. Then a basic mining algorithm, a top-*k* mining algorithm and various optimized methods are presented.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 287–300, 2013.

The rest of the paper is organized as follows. Section 2 introduces related works. Section 3 considers the time aspects of the data records. Section 4 defines related concepts and proves the anti-monotonicity of prevalent co-location patterns with time constraints. In Section 5, we develop two basic mining algorithms, and produce optimizing strategies for the two mining algorithms. We present experimental analysis in Section 6, and conclusions in Section 7.

2 Related Works

A large amount of research has been made into co-location pattern mining in recent years, over both certain and uncertain data sets.

For the research on certain data, there are many algorithms, such as the join-based algorithm based on the joining operation [1], the join-less algorithm based on star table instances' pruning [2], the CPI-tree algorithm based on the instances' neighbor relationship tree [3], the order-clique-based algorithm based on ordered cliques [4], and the density-based algorithm based on the partition of spatial instances [5]. To deal with the situation where there exists a rare feature in the data sets (i.e. the number of instances with this feature is significantly smaller than those with other features), paper [6] proposed a maximal-participation-ratio-based algorithm, while paper [7] provided a minimum-weighted-participation-ratio-based algorithm. As it is not easy to specify the thresholds of distance and prevalence, paper [8] proposed a novel iterative mining framework that discovers spatial co-location patterns without predefined thresholds. In a similar manner to traditional association rule mining with both maximum frequent itemsets mining and closed frequent itemsets mining, paper [9] proposed a top-*k* closed co-location patterns' concept with corresponding mining algorithms. Paper [10] researched maximum co-location pattern mining, and further promoted co-location pattern mining applications.

For uncertain data, Lu et al [11] proposed an UJoin-based algorithm to mine prevalent co-locations. Wang et al [12] discovered co-location patterns from uncertain data sets with interval data. Paper [13] took fuzzy features into consideration, defined a fuzzy participation index and proposed a co-location pattern mining algorithm with fuzzy features. Paper [14] considered the uncertainty of the existence of a spatial instance in a possible world model, and for this model defined probabilistic prevalent co-locations, and provided exact and approximation algorithms to mine the probabilistic prevalent co-location patterns.

Thus there are many techniques for spatial co-location pattern mining, but not for spatial data sets with time constraints. This paper explores how to extract spatial co-location patterns from historic spatial data where there are time constraints.

3 Approaching Historic Data

In many research situations there are historic records, sometimes very accurate, often not so. Additionally, factors such as climate change mean that in some situations there may be uncertainty about how much the past is relevant to the present. To accommodate both these reasons we have adopted a net present value approach to

historic records, whereby the importance of evidence from the past is progressively diminished as time has gone by. Using the **Net Present Value** (**NPV**) technique normally applied to financial transactions and available in all good spreadsheet programs, we supply a discount factor to the presence or absence of a feature in the historical record. Table 1 shows the effect, relating to the spatial presence of a particular plant species:

Table 1. The effect, relating to the spatial presence of a particular plant species

5% discount	NPV	now	last year	previous	previous	previous	previous	previous	previous	previous	previous	description
1	8.11	1	1	1	1	1	1	1	1	1	1	persistent
2	2.72			1	1	1						in the past only
3	1.95	1			1							interrupted record
4	1.00	1										new

In the rest of this paper the NPV values of time have been rounded to integer values for clarity in the diagrams and equations, and are known as NPV-times.

4 Definitions and a Lemma

This section firstly gives the basic concepts of spatial co-location patterns with NPV-time constraints. The important downward closure property of prevalent co-location patterns with their NPV-time constraints is then proved.

Definition 1. *Spatial features and spatial instances with time constraints*
A spatial feature f_i presents a kind of spatial object types in a spatial area. The *j*-th **spatial instance with NPV-time constraint** of the spatial feature f_i is denoted as $f_i.j^t$, where *t* presents the NPV-time from the historical records of the feature.

For example, in a geographical area one type of tree (a typical feature), A, could have instances $A.1^3$, $A.2^{10}$, denoting the 1st and 2nd instance of A, where the net present values of their existence times are 3 and 10 years respectively.

Definition 2. *Spatial neighbor relationship R*
When two spatial instances satisfy the condition that their Euclidean distance is less than or equal to a user-given distance threshold *dist_threshold*, we say they satisfy the **spatial neighbor relationship *R***.

Definition 3. *R Clique*
For a spatial instance set $I = \{i_1^{t_1}, i_2^{t_2}, \cdots, i_m^{t_m}\}$, if there are $\{R(i_j^{t_j}, i_k^{t_k}) \mid 1 \le j \le m, 1 \le k \le m\}$, then *I* is called an ***R* clique**.

Definition 4. *Co-location patterns c*
A **co-location pattern *c*** is a subset of spatial features. The length of a co-location pattern ***c*** is called the ***size*** of *c*, i.e. *size* (*c*)=|*c*|.

Definition 5. *Row instances*
If an *R* clique *I* contains all the features of a co-location pattern *c*, and no proper subset of *I* does so, then the *I* is called a **row instance** of *c*, denoted as *row_instance* (*c*).

Definition 6. *Weight wr* ($f_i.j^{t_i}$) *of instance* $f_i.j^{t_i}$ *in a row instance*

If a row instance has *n* instances, the **weight of an instance** $f_i.j^{t_i}$ **in the row instance** is defined as the ratio of the minimum NPV-time in all instances of the row instance to the NPV-time of **instance** $f_i.j^{t_i}$, i.e., $wr(f_i.j^{t_i}) = \min\{t_1, t_2, \ldots t_n\}/t_i$.

For example, if a row instance of a co-location pattern {*A, B, C*} is {$A.1^3$, $B.1^5$, $C.2^2$}, then in this row instance, the weight of instance $A.1^3$ is $wr(A.1^3)$=min{3, 5, 2}/3 =2/3.

Definition 6 is reasonable because the longer is the NPV-time of instances the bigger is the applied weight and the longer an instance exists in isolation the smaller is its applied weight.

Definition 7. *Table instance*
The collection of all row instances of a co-location pattern *c* forms the **table instance** of *c*, *table_instance* (*c*).

Definition 8. *Weight wt* ($f_i.j^{t_i}$) *of instance* $f_i.j^{t_i}$ *in table instance*

In a table instance, if an instance $f_i.j^{t_i}$ appears in *k* row instances, the **weight of this instance in the table instance** is defined as the maximum of the weights of this instance in all row instances, i.e. $wt(f_i.j^{t_i}) = \max\{wr_1(f_i.j^{t_i}), \cdots, wr_k(f_i.j^{t_i})\}$.

Definition 9. *Weighted participation ratio*
WPR(*c*, f_i) represents the **weighted participation ratio** of feature f_i in a *k*-size co-location pattern *c*, and is defined as the ratio of the sum of all weights *wt* （ $f_i.j^{t_i}$ ） of the instances of feature f_i in *table_instance* (*c*) and all instances of f_i, i.e.,

$$\mathrm{WPR}(c, f_i) = \frac{\sum_{f_i.j^{t_i} \in \prod_{f_i}(table\_instance(c))} wt(f_i.j^{t_i})}{|\,\mathrm{table\_instance}(\{f_i\})\,|}$$

Definition 10. *Weighted participation index*
WPI(*c*) represents the **weighted participation index** of a co-location pattern $c=\{f_1,\cdots,f_k\}$, defined as the minimum of the weighted participation ratio of all features in *c*, i.e., $\mathrm{WPI}(c) = \min_{i=1}^{k}\{\mathrm{WPR}(c, f_i)\}$.

When WPI(*c*) is greater than or equal to a user specified minimum weighted participation index threshold ***Wmin_prev***, we say the co-location pattern *c* is ***prevalent***. Prevalent co-location patterns discovered from spatial data with NPV-time constraints are called ***prevalent co-locations with time constraints***.

Example 1. In Fig. 1, there are four spatial features, *A*, *B*, *C*, *D*. Their instances respectively are $A=\{A.1^2, A.2^5, A.3^8, A.4^2\}$, $B=\{B.1^4, B.2^3, B.3^6\}$, $C=\{C.1^6, C.2^1, C.3^2\}$, $D=\{D.1^3, D.2^5\}$. A point in this figure represents a spatial instance. A solid line between two points indicates a spatial neighbor relationship *R* between two instances. If *Wmin_prev* is given as 0.4, we determine whether co-location pattern $c=\{C, D\}$ is prevalent or not.

Firstly, the table instance of *c* is $\{\{C.1^6, D.1^3\}, \{C.3^2, D.1^3\}, \{C.2^1, D.2^5\}\}$. $wt(C.1) = max\{min\{6,3\}/6\}=1/2$, $wt(C.3)=max\{min\{2,3\}/2\}=1$, $wt(C.2)=max\{min\{1, 5\}/1\}=1$, $wt(D.1) = max\{min\{6, 3\}/3, min\{2, 3\}/3\} =1$, $wt(D.2) = max\{min\{1, 5\}/5\}=1/5$. Secondly, the weighted participation ratios of features in *c* are respectively $WPR(c,C)=(1/2+1+1)/3=5/6$, $WPR(c,D)=(1+1/5)/2=0.6$. Thus, the weighted participation index of *c* $WPI(c)=min\{5/6, 0.6\}=0.6$. The value $WPI(c)$ is greater than *Wmin_prev*, so the co-location pattern $c=\{C, D\}$ is prevalent.

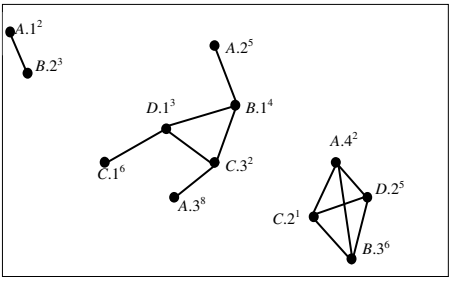


Fig. 1. An example of spatial data with NPV-time constraints

Lemma 1. *The weighted participation ratio (WPR) and the weighted participation index (WPI) both monotonically decrease as the size of the co-location pattern increases.*

Proof. If a spatial instance of a feature is contained in row instances of a co-location pattern *c,* for a co-location pattern $c' \subseteq c$, this instance must be contained in row instances of c' and the weight of this instance in *c* must be less than or equal to the weight of this instance in c' due to Definition 6 and Definition 8. In other words, the weighted participation ratio monotonically decreases as the size of the co-location pattern increases.

Suppose $c=\{f_1,\ldots,f_k\}$, then $WPI(c \cup f_{k+1}) = \min_{i=1}^{k+1}\{WPR(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{WPR(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{WPR(c, f_i)\} = WPI(c)$. Therefore, the WPIs of co-location patterns also monotonically decrease as the size of the co-location patterns increase.

Based on Lemma 1, we know that if a *k*-size co-location pattern is prevalent, any *k*-1-size sub-pattern of it is also prevalent.

5 Algorithm Design

Firstly, we give a basic algorithm to mine all prevalent co-locations with NPV-time constraints, and then an optimized computation is proposed. Secondly, due to the difficulty of determining a sensible *Wmin_prev*, a top-*k* spatial co-location mining algorithm is designed.

5.1 Basic Mining Algorithm

Based on expanding the classical join-based algorithm, we designed Algorithm 1 to mine all prevalent co-locations with NPV-time constraints. The basic algorithm is called the w-join-based algorithm since the core work in this algorithm is to compute the WPI of candidate co-locations with NPV-time constraints.

Algorithm 1. w-join-based algorithm

Input:

Spatial feature sets: $F=\{f_1,\ldots,f_n\}$;
Spatial instance set with NPV-time constraints: S;
Distance threshold of spatial neighbor relationship: *dist_threshold*;
Minimum weighted participation index threshold: *Wmin_prev*;

Output:

The set of co-location patterns whose WPI is larger than or equal to *Wmin_prev*: *WP*

Variables:

k: the size of the co-location patterns with NPV-time constraints;
C_k: the k-size candidate co-locations' set with NPV-time constraints;
T_k: the co-location table instance set with NPV-time constraints in C_k;
P_k: the k-size prevalent co-locations' set with NPV-time constraints;

Steps:

(1) $P_1=F$, $WP=P_1$; // 1-size co-locations are all prevalent;
(2) Calculating spatial neighbor relationship R between spatial instances;
(3) for (k =2; $P_{k-1} \neq \emptyset$; k++) {
3.1 C_k =gen_cand_co-location (k, P_{k-1}); //generating candidate co-locations;
3.2 T_k =gen_table_instance (C_k, T_{k-1}); //calculating table instances;
3.3 P_k =select_prevalence_co-location (C_k,T_k,*Wmin_prev*);
//obtaining k-size prevalent co-locations with NPV-time constraints;
3.4 $WP \leftarrow WP \cup P_k$; }
//all prevalent co-locations with NPV-time constraints.
(4) return *WP*

Explanation of Algorithm 1: Step (1) initializes two variables P_1 and *WP*; step (2) calculates the spatial neighbor relationship R according to the given distance threshold *dist_threshold*; step (3) iteratively generates the set of prevalent co-location patterns with NPV-time constraints, where step 3.1 generates candidate patterns, step 3.2 computes table instances of candidate patterns, step 3.3 calculates a weighted

participation index according to the related definitions of co-location patterns with NPV-time constraints, and then obtains prevalent co-location patterns with NPV-time constraints less than *Wmin_prev*; step (4) returns all prevalent co-locations with NPV-time constraints.

Analysis of Algorithm 1: Like the classical join-based algorithm, as the size of co-locations with NPV-time constraints increases, the joining operation for generating table instances becomes quite time-consuming. Worse, Algorithm 1 has to compute the weight of instances in row instances and table instances after calculating the table instance of a candidate. So in dense data sets, the basic mining algorithm is not an ideal method. An optimized method is presented in the next sub-section to improve the performance of the basic mining algorithm.

5.2 Optimized Computation of Candidates

The optimized method is based on Lemma 2 and 3.

Lemma 2. *For a feature in a k-size co-location with NPV-time constraints, if its weighted participation ratio is less than Wmin_prev, then this k-size pattern and any of its super-patterns cannot be prevalent NPV-time constrained co-location patterns.*

Proof: According to Definition 10, the weighted participation index is the minimum of all the feature's weighted participation ratios. If the weighted participation ratio of one feature in a *k*-size co-location pattern is less than *Wmin_prev*, the weighted participation index of this pattern must be less than *Wmin_prev*. So this *k*-size pattern cannot be prevalent. According to Lemma 1, as the size of the co-location patterns with NPV-time constraints increase, their weighted participation ratio decreases. So the super-patterns of this co-location pattern cannot exist in prevalent co-locations with NPV-time constraints. Thus the lemma is proved.

Generally, the non-prevalent co-location patterns' set is larger than the prevalent set. In order to terminate the calculation of a pattern as soon as possible, we give Lemma 3.

Lemma 3. *The weighted participation ratio of a feature in a co-location pattern with NPV-time constraints monotonously decreases as the NPV-time of instances of the feature increases.*

Proof: According to Definition 6, we know that the larger is the NPV-time of an instance, the smaller is the weight of the instance in its row instance. According to Definition 8 and Definition 9, we can infer that as the NPV-time of instances increases, the weighted participation ratio of the corresponding pattern monotonously decreases. In other words, as the NPV-time of an instance gets larger, the weighted participation index of a corresponding pattern does not exceed the original value.

Applying Lemma 3 in practice, we compute the average NPV-time of features' instances. The weighted participation ratio of features in a candidate is computed from the descending order of the average NPV-time of features' instances. From Lemma 3 we can infer that the larger is the average NPV-time of a feature's instances

in a co-location pattern, the smaller will be its weighted participation ratio. The algorithm applying the optimized procedure is called the **p2-w-join-based** algorithm to distinguish it from the **w-join-based** algorithm. The optimized computation procedure of candidates is:

Procedure Optimized calculation of candidates

Steps:

(1) For each feature of a candidate co-location, the average NPV-time of instances of this feature is calculated. The weighted participation rate of features are computed in the descending order of their average NPV-time;

(2) If the weighted participation rate of a feature in candidate is less than *Wmin_prev*, remove this candidate from the candidate set and terminate the calculation of this candidate; Otherwise, go to step (3);

(3) If the weighted participation rates of all features in this candidate have been calculated, compute the weighted participation index of this candidate, and then go to step (4); Otherwise, calculate the weighted participation rate of the next feature, and then go to step (2);

(4) If the weighted participation index of this candidate is greater than or equal to *Wmin_prev*, put it into the set of prevalent co-location patterns with NPV-time constraints;

5.3 Top-*k*-w Mining Algorithm

The basic mining algorithm presented above requires users to specify in advance the thresholds of distance and minimum weighted participation index. In practice, however, it is not easy to specify suitable thresholds, especially for the minimum weighted participation index threshold *Wmin_prev*. In this section we introduce the concept of top-*k* prevalent co-location patterns with NPV-time constraints, which can effectively avoid the specification of the threshold *Wmin_prev*.

Definition 11. *Top-k prevalent co-location patterns with NPV-time constraints*
The *k* co-location patterns discovered from spatial data with NPV-time constraints, which have highest weighted participation index (WPI), are called the **top-*k* prevalent co-location patterns with NPV-time constraints**, where *k* is specified by user.

Top-*k* mining not only replaces the parameter *Wmin_prev* with the parameter *k* which is much easier to specify, but it also ensures that co-locations with the highest prevalence are output first. We now give a lemma for dynamically finding the top-*k* prevalent co-locations.

Lemma 4. *Any co-locations whose subsets do not exist in the top-k prevalent co-locations are not in the top-k prevalent co-locations with NPV-time constraints.*

Proof: According to Lemma 1, the WPI of co-locations monotonically decreases as the size of co-locations increases. So, if a co-location occurs in the result of the top-*k*

prevalent co-locations with NPV-time constraints, the WPI of its any subset cannot be less than the WPI of this co-location, i.e., its subsets should occur in the mining results.

Based on Lemma 4, we design Algorithm 2 to mine the top-k prevalent co-location with NPV-time constraints. In a size-l iteration of Algorithm 2 (Step (4)), each size-l candidate co-locations are inserted into *topkP* (the queue of the top-k prevalent co-locations with their WPIs) after their WPI are computed, and then the size-(l+1) candidates are generated based on size-l co-locations in *topkP*. Any co-locations whose subsets do not exist in *topkP* can safely be ignored.

Algorithm 2. Top-k-w algorithm

Input:

Spatial feature sets: $F=\{f_1,f_2,\ldots f_n\}$;

The set of spatial instances with NPV-time constraint: S;

Distance threshold of spatial neighbor relationship: *dist_threshold*;

The k in the top-k prevalent co-locations with NPV-time constraints;

Output:

The queue of the top-k prevalent co-locations with their WPIs: *topkP;*

Variables:

l: the size of the co-location patterns with NPV-time constraint;

C_l: the l-size candidate co-locations' set;

T_l: the co-location table instances' set in C_l;

P_l: the set of the l-size prevalent co-locations with their weighted participation index ($<c,WPI(c)>$);

Steps:

(1) Calculating the spatial neighbor relationships R between spatial instances with NPV-time constraints;

(2) Generating 2-size candidate co-locations C_2;

(3) l=2;

(4) **WHILE** C_l <> null **Do**

4.1 **For** each c in C_l

{ compute the value WPI(c);

insert ($<c$,WPI(c)>) into the QUEUE *topkP* sorted by the value WPI in descending order; **}**

4.2 l=l+1;

4.3 generate l-size candidate co-locations C_l based on *topkP*;

ENDWHILE

(5) Return *topkP*

In implementing Algorithm 2, we encounter the problem that the WPI of the higher size candidate may equal the value of *topkP*[k].WPI. Which one in the *topkP* should be replaced by this higher sized one? We know that the information content of a higher pattern contains that of all its sub-patterns, so we adopt as a method that the smallest WPI pattern in all sub-patterns is replaced by this higher size one.

For example, in a top-4 co-location mining, suppose there were four co-locations in the queue *topkP,* <AB,0.80>, <AC,0.75>, <BC,0.70> and <CD,0.60>. If WPI({ABC})=0.60, then {BC} is replaced by {ABC}, and the new queue of the top-4 prevalent co-locations is {<AB,0.80>, <AC,0.75>, <ABC,0.60>, <CD,0.60>}.

In computing the value WPI(*c*), the optimized strategy based on Lemma 2 and lemma 3 presented in Section 4.2 can be used. That is, after there are *k* patterns in *topkP*, the weighted participation rate of features can be computed in the descending order of their average NPV-time. The features which might have the minimum weighted participation rate can be calculated first. If the weighted participation rate of this feature is less than *topkP*[*k*].WPI, the calculation of candidate *c* is terminated prematurely. The algorithm applying the optimized strategy is called the **p-top-*k*-w** algorithm.

6 Experimental Analysis

In order to analyze the correctness, rationality and efficiency of the proposed algorithms and optimized methods, a series of experiments have been performed on "real+synthetic" data sets. Our algorithms are implemented in C# and executed on an Intel core i3 CPU with 4GB of memory and 64-bit Windows 7 operating system.

6.1 Data Sets

The data sets are setup for evaluating the algorithms. In the data sets, 14 plant species (spatial features) and 1400 instances are selected randomly from the plant data sets of "Three Parallel Rivers of Yunnan Protected Areas". The distribution area of spatial instances are standardized in a [0,500] × [0,500] space, while the NPV-time constraints of instances are generated by a synthetic method. In the following experiments, we measure the run time and the results of the algorithms under different parameter settings. Table 2 summarizes the parameters and their default values.

Table 2. Parameter settings

Parameter	Default value
dist_threshold	50
Wmin_prev	0.4
The *k* in top-*k*	20
Number of spatial features	14
Total number of spatial instances	1400

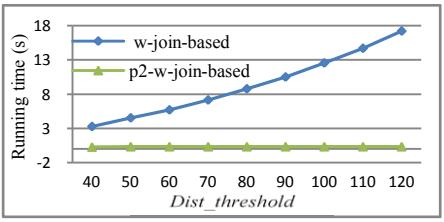
6.2 Algorithm Performance Evaluation

In this subsection, we compare the performance of the basic mining algorithm (w-join-based) against its optimized algorithm (P2-w-join-based), and also of the top-*k* mining algorithm (top-*k*-w) with the optimized algorithm (P-top-*k*-w).

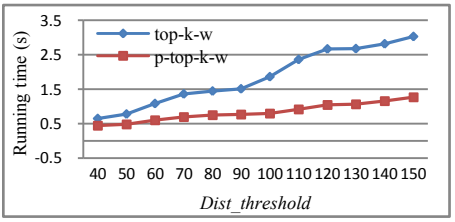
6.2.1 Effect of *Dist_Threshold*

The run time of w-join-based algorithm and p2-w-join-based algorithm as a function of *dist_threshold* is shown in Fig. 2(a). The results show that the run time increases as *dist_threshold* increases. This is because increasing *dist_threshold* makes the neighbor areas larger, so increasing the number of neighbor relationships. We can also see that the larger is *dist_threshold*, the greater the difference between the two algorithms. It illustrates that the pruning and optimization strategies work better as *dist_threshold* is increased.

Fig. 2(b) is the run time of top-*k*-w and p-top-*k*-w when *dist-threshold* is varied from 40 to 150. These results also show that optimized method is better than the basic algorithm. Note that the cost of the top-*k* mining algorithms lies between w-join-based and p2-w-join-based and that the cost of the p2-w-join-based algorithm is the smallest of all the algorithms using the parameter settings of Table 1.



(a) w-join-based and p2-w-join-based

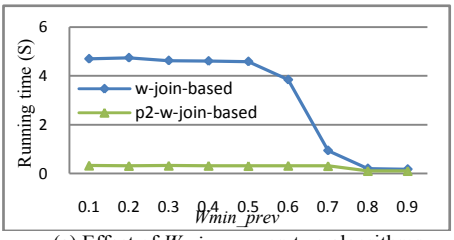


(b) top-*k*-w and p-top-*k*-w

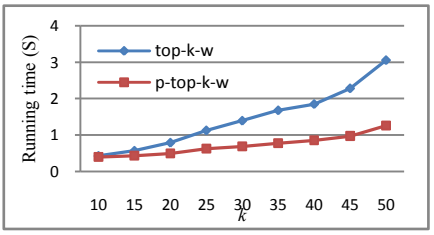
Fig. 2. Effect of dist_threshold on the algorithms

6.2.2 Effect of *Wmin_Prev* or *k*

For the basic mining algorithm w-join-based and its optimized algorithm P2-w-join-based, *Wmin_prev* is the most important parameter. The values of *Wmin_prev* not only determine the mining results, but also affect the cost of algorithms. Fig. 3(a) shows the impact of *Wmin_prev* on the run time of the basic mining algorithm and the optimized mining algorithm. As can be observed from Fig. 3(a), the larger *Wmin_prev* is, the faster the algorithms are. It can also be seen that the optimized algorithm is always more efficient than the basic algorithm.



(a) Effect of *Wmin_prev* on two algorithms



(b) Effect of *k* on top-*k*-w and p-top-*k*-w

Fig. 3. Effect of *Wmin_prev* or *k* on the algorithms

Fig. 3(b) shows the impact of k on the top-k mining algorithms. The run time of the two algorithms increases linearly with increases in k. The larger k is, the greater is the difference between them.

6.2.3 Effect of the Number of Instances

We now test the scalability of our algorithms by varying the total number of spatial instances from 1K to 5K. Results are shown in Fig. 4 with an exponential scale, from which we can see that the performances of all algorithms degrade as the dataset grows. However, the optimized algorithms are consistently superior to the basic method and the optimized mining algorithm p2-w-join-based is consistently the best one when using the parameter settings of Table 1.

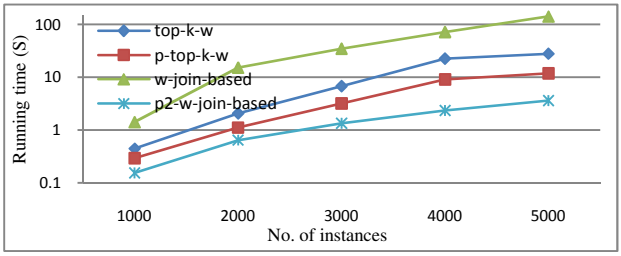
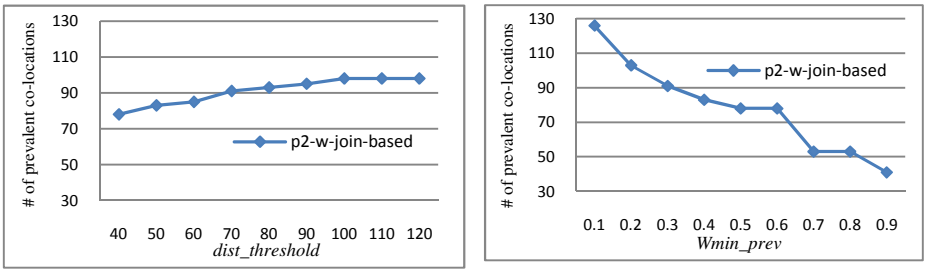


Fig. 4. Effect of the number of instances on the algorithms

6.3 Analyzing the Number of Prevalent Co-locations

In this subsection, we study the impact of the parameters on the number of prevalent co-location patterns with NPV-time constraints generated by p2-w-join-based. We note that the result of the w-join-based is the same as that of p2-w-join-based, and that the result of the top-k mining algorithms is always k patterns.

Fig. 5(a) shows the number of the prevalent co-locations mined by the p2-w-join-based technique where the parameter *dist-threshold* is varied from 40-120. We can see that the number of prevalent patterns grows as *dist-threshold* increases. Since the



(a) Effect of *dist_threshold* (b) Effect of *Wmin_prev*

Fig. 5. Effect of *dist_threshold or Wmin_prev* on mining results

greater is *dist_threshold*, the larger is the neighbor area, there may be a growth in the prevalence of a co-location. The impact of the parameter *Wmin_prev* on the number of prevalent co-locations is shown in Fig. 5(b). It is clear that the number of prevalent patterns grows as *Wmin_prev* decreases. Comparing Fig. 5 (a) to 5 (b), we can see that the impact of the parameter *Wmin_prev* on the mined results is larger than that of *dist_threshold*. That is why we have concentrated on the top-*k* mining approach in this paper.

7 Conclusion

Although time of occurrence is an important attribute of spatial objects, the time attribute in spatial co-location pattern mining has not previously been researched. In this paper we have studied this problem by dealing with the NPV-time attribute as a weight of spatial data, defining the *weight participation ratio*, *weight participation index*, and *prevalent co-locations with NPV-time constraints.* We have developed a basic mining algorithm, a top-*k* mining algorithm and also optimized algorithms to mine the prevalent co-locations with NPV-time constraints. Extensive experiments have shown that our optimized algorithms consistently achieve superior performance to the baseline approaches. The next step will be further application studies.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (No. 61063008, No. 61272126, No. 61262069), the Science Foundation of Yunnan Province (No. 2010CD025) and the Research Foundation of the Educational Department of Yunnan Province (No. 2012C103).

References

1. Huang, Y., Shekhar, S., Xiong, H.: Discovering co-location patterns from spatial data sets: a general approach. IEEE Transactions on Knowledge and Data Engineering (TKDE) 16(12), 1472–1485 (2004)
2. Yoo, J.S., Shekhar, S.: A join-less approach for co-location pattern mining: a summary of results. IEEE Transactions on Knowledge and Data Engineering (TKDE) 18(10), 1323–1337 (2006)
3. Wang, L., Bao, Y., Lu, J., Yip, J.: A new join-less approach for co-location pattern mining. In: The 8th IEEE International Conference on Computer and Information Technology (CIT 2008), pp. 197–202. IEEE Press, New York (2008)
4. Wang, L., Zhou, L., Lu, J., Yip, J.: An order-clique-based approach for mining maximal co-locations. Information Sciences 179(19), 3370–3382 (2009)
5. Xiao, X., Xie, X., Luo, Q.: Density-based co-location pattern discovery. In: The 16th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems, Irvine, California, USA, pp. 11–20. ACM Press (2008)
6. Huang, Y., Pei, J., Xiong, H.: Mining co-location patterns with rare events from spatial data sets. Geoinformatica 10(3), 239–260 (2006)

7. Feng, L., Wang, L., Gao, S.: A new approach of mining co-location patterns in spatial datasets with rare features. Journal of Nanjing University (Natural Sciences) 48(1), 99–107 (2012)
8. Qian, F., He, Q., Chiew, K., He, J.: Spatial co-location pattern discovery without thresholds. Knowledge Information System (2012)
9. Yoo, J.S., Bow, M.: Mining Top-k Closed Co-Location Patterns. In: The IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM 2011), pp. 100–105. IEEE Press, New York (2011)
10. Yoo, J.S., Bow, M.: Mining Maximal Co-Located Event sets. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part I. LNCS, vol. 6634, pp. 351–362. Springer, Heidelberg (2011)
11. Lu, Y., Wang, L., Zhang, X.: Mining frequent co-location patterns from uncertain data. Journal of Frontiers of Computer Science and Technology 3(6), 656–664 (2009)
12. Wang, L., Chen, H., Zhao, L., Zhou, L.: Efficiently Mining Co-location Rules on Interval Data. In: Cao, L., Feng, Y., Zhong, J. (eds.) ADMA 2010, Part I. LNCS, vol. 6440, pp. 477–488. Springer, Heidelberg (2010)
13. Ouyang, Z., Wang, L., Chen, H.: Mining spatial co-location patterns for fuzzy objects. Chinese Journal of Computers 34(10), 1947–1955 (2011)
14. Wang, L., Wu, P., Chen, H.: Finding Probabilistic Prevalent Co-locations in Spatially Uncertain Data Sets. IEEE Transactions on Knowledge and Data Engineering TKDE 25(4), 790–804 (2013)

Mining Co-locations from Spatially Uncertain Data with Probability Intervals

Lizhen Wang, Peng Guan, Hongmei Chen*, and Qing Xiao

Department of Computer Science and Engineering, School of Information Science and Engineering, Yunnan University, Kunming 650091, China
{lzhwang,hmchen}@ynu.edu.cn

Abstract. Uncertain data are inherent in many applications, and are usually described by precise probabilities. However, it is difficult to obtain precise probabilities over uncertain data in applications. This paper studies the problem of mining co-locations from spatially uncertain data with probability intervals. Firstly, it defines the possible world model with probability intervals, and proves that probability intervals of all possible worlds are feasible. Secondly, based on the feasible probability interval, it converts the probability intervals of possible worlds into point probabilities. Further, it defines the related concepts of probabilistic prevalent co-locations. Thirdly, it gives two lemmas for optimizing the computation of prevalence point probability of a candidate co-location. Further, it proves the closure property of prevalence point probability. Finally, the experiments on synthetic and real data sets show that the algorithms are effective and significant.

Keywords: Uncertain data mining, probability interval, possible world, probabilistic prevalent co-location.

1 Introduction

Spatial co-location patterns are subsets of spatial features, and mining co-locations is to discover correlation and relationships between features. For example, West Nile virus usually appears in areas where mosquitoes are abundant and poultry are kept; botanists find that 80% of sub-humid evergreen broadleaved forests grow with orchid plants [1].

Uncertain data are inherent in many applications, and are usually described by precise probabilities. For example, an object exists in a location with the probability 0.4. In practice, however, it is difficult to obtain precise probabilities, due to the probabilistic description may come from either inaccurate estimation by the experts or inconsistent assertions of different experts. For example, the experts maybe say the probability is between 0.5 and 0.7; or some experts say the probability is about 0.5, and other say it is about 0.7. In these cases, the obtained probabilities are not precise.

Although several methods of modeling imprecise probabilities have been proposed [2], research on mining co-locations from spatially uncertain data with imprecise

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 301–314, 2013.

probabilities is still missing. As an exploration in this field, this paper discusses how to find co-locations from spatially uncertain data with probability intervals.

Generally, the main contributions of the paper can be summarized as follows:

--Firstly, the possible world model with probability intervals is defined, and that probability intervals of all possible worlds are feasible is proved.

--Secondly, based on the feasible probability interval, the probability intervals of possible worlds are converted into point probabilities. Further, the related concepts of probabilistic prevalent co-locations are defined.

--Thirdly, in order to optimize the computation of prevalence point probability of a candidate co-location, two lemmas are given. Further, the closure property of the prevalence point probability is proven, and an improving algorithm is proposed.

--Finally, the experiments on synthetic and real data sets show that the algorithms are effective and significant.

The rest of the paper is organized as follows. Section 2 introduces related works, and Section 3 gives related concepts. Section 4 defines the possible world model with probability intervals and the related concepts of probabilistic prevalent co-locations. Section 5 and 6 present the algorithms and experiments respectively, and Section 7 concludes the paper.

2 Related Works

Mining co-locations is an important research area in spatial data mining. Paper [3] provided related definitions and proposed the classic Join-Based algorithm. Extensive researches have been made over certain and uncertain data in recent years [4-12].

In terms of certain data, paper [4] proposed a star-neighborhood-based Join-Less algorithm; paper [5] put forward the CPI-tree algorithm which uses tree-structure to store the neighbor relationships between instances, and the table instances of candidate co-locations can be fast generated by CPI-tree; Paper [6] provided an Order-Clique-Based algorithm, which is used for mining maximal prevalent co-locations, avoiding both outputting a large amount of smaller-size prevalent patterns and storing a mass of table instances; Paper [7] took fuzzy features into consideration, defining the fuzzy participation index and proposing the algorithm with fuzzy features; To deal with the situation where there exists a rare feature (i.e. the number of instances with this feature is significantly smaller than those with other features), paper [8] proposed a Maximal-Participation- Ratio-Based algorithm. If there is a rare feature but some candidate patterns do not contain the rare feature, utilizing the algorithm in [8] may lead to some non-prevalent patterns being mined. To cope with this problem, paper [9] defined the minimum weighted participation ratio and gave an algorithm based on this ratio. This algorithm can not only mine the prevalent co-locations with rare features, but also exclude the non-prevalent patterns.

In terms of uncertain data, paper [10] utilized probability density function to describe the uncertainty of spatial instances' locations, defined the expected distance between instances and proposed the UJoin-Based algorithm; paper [11] considered the uncertainty of spatial instances' existence under the possible world model, defined

the minimum probability participation ratio, modified the Order-Clique-Based algorithm in paper [6] and proposed the U-Order-Clique-based algorithm; Paper [12] also considered the uncertainty of spatial instances' existence, defined the prevalence probability and expected participation index based on the possible world model, and provided exact and approximation algorithms that mines probabilistic prevalent co-location patterns. All these algorithms are used to mine co-locations from spatially uncertain data with precise probability values.

Distinct from the above research, this paper deals with spatially uncertain data with probability intervals, and explores how to mine co-locations in this case.

3 Related Concepts

3.1 Co-locations and Spatially Uncertain Data with Probability Intervals

A Spatial feature f represents different kinds of things in an area. For example, a plant species is a feature. An object of f at a location is called an **instance** of f, e.g. a plant is an instance of the plant species. We use the **spatial neighbor relationship** R to describe the relationships between instances. When considering the distances between instances, we say that two instances a and b satisfy R, denoted as R(a,b), if $distance(a,b) \leq d$, where d is a threshold.

Spatial co-location pattern c is a subset of the features' set F. The number of features in c is called the size of c. For F={A,B,C}, {A,B} is a co-location with size 2. If there is a set of instances $I=\{i_1,i_2,\ldots,i_m\}$ such that $\{R(i_j,i_k) | 1 \leq j \leq m, 1 \leq k \leq m\}$, then I is called a **R Clique.** If I contains all the features in c, but there is no proper subset of I containing all the features in c, then I is a **row instance** of c. The set of all row instances of c is called the **table instance** of c, denoted as $table\_instance(c)$.

In spatially uncertain data with probability intervals, the existence of an instance a is uncertain, i.e. it may exist or not. Further, the probability of the existence p_in(a) is imprecise, and expressed by the **probability interval** $[L_a, U_a]$, where $0 \leq L_a \leq U_a \leq 1$. Then, the probability of the non-existence !p_in(a) can be expressed by $[1-U_a, 1-L_a]$.

Exampe 1. Fig. 1 describes the instances in the features' set F={A,B,C}. For example, A.1 denotes the first instance of feature A, and p_in(A.1)=[0.1,0.3], !p_in(A.1) =[0.7,0.9]. The line between A.1 and B.1 mean R(A.1,B.1). The table instance of c={A,B} is {{A.1,B.1},{A.2,B.1},{A.2,B.2}}.

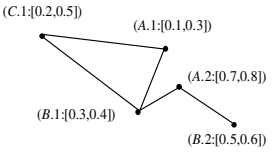


Fig. 1. The instances in the features' set $F=\{A,B,C\}$

3.2 The Feasible Probability Interval

The paper will utilize the feasible probability interval as the bridge to mine co-locations from spatially uncertain data with probability intervals.

Definition 1[13]. *For a n-dimensional probability interval n-PRI(L,U)={[L_i,U_i]| $0 \le L_i \le U_i \le 1$, $i=1,2,\ldots,n$} on the set of basic events $\Omega=\{w_1,w_2,\ldots,w_n\}$, where $L_i \le p(w_i) \le U_i$, it is said to be* ***reasonable*** *if it satisfies that* $\sum_{i=1}^{n} \boldsymbol{L_i} \le \mathbf{1}$, $\sum_{i=1}^{n} \boldsymbol{U_i} \ge \mathbf{1}$.

Definition 2[13]. *For a reasonable n-PRI(L,U), it is called* ***feasible*** *if it satisfies that* $\sum_{i=1,i\neq j}^{n} \boldsymbol{L_i} + \boldsymbol{U_j} \le \mathbf{1}$, $\sum_{i=1,i\neq j}^{n} \boldsymbol{U_i} + \boldsymbol{L_j} \ge \mathbf{1}$, *for all j=1,2,…,n.*
A reasonable *n*-PRI(*L*,*U*) implies that there exists $\{p_i | L_i \le p_i \le U_i, i=1,2,\ldots,n\}$ satisfying $\sum_{i=1}^{n} p_i$. A feasible *n*-PRI(*L*,*U*) implies that, for a p_j, $L_j \le p_j \le U_j$, $j=1,2,\ldots,n$, there exists $\{p_i | L_i \le p_i \le U_i, i=1,2,\ldots,j\text{-}1,j+1,\ldots,n\}$ satisfying $\sum_{i=1,i\neq j}^{n} p_i + p_j = 1$.

Example 2. Let a 3-PRI(*L*,*U*) be {[0.1,0.3], [0.3,0.6], [0.2,0.5]}. According to Definition 1 and 2, this 3-PRI(*L*,*U*) is reasonable and feasible.

4 Co-locations in Spatially Uncertain Data with Probability Intervals

4.1 The Possible World Model with Probability Intervals

The possible world model is the most widely used model in uncertain data mining. Given a spatially uncertain data set with probability intervals, the probabilities of possible worlds are also imprecise, and can be also expressed by probability intervals.

Definition 3. *Let W_F be the set of all possible worlds generated by the instances' set S of the features' set F. For a possible world $w_F \in W_F$, its probability interval p_in(w_F) can be calculated by:* $p\_in(w_F) = \prod_{x \in w_F} p\_in(x) \times \prod_{y \in S - w_F} !p\_in(y)$

where the operation $\times$ and $\prod$ both represent the multiplying operation of intervals. For intervals $IA=[a_1,a_2]$ and $IB=[b_1,b_2]$, where $a_1,a_2,b_1,b_2 \ge 0$, $IA \times IB=[a_1b_1,a_2b_2]$.

Example 3. Taking Fig. 1 as an example, we can calculate the probability interval of the possible world w_F={*A*.1,*B*.1,*B*.2,*C*.1} according to Definition 3:
p_in(w_F)=p_in(A.1)×p_in(B.1)×p_in(B.2)×p_in(C.1)×!p_in(A.2)=[0.0006,0.0108]

For *n* instances, there are 2^n possible worlds in total, and their probability intervals constitute a 2^n-dimensional probability interval 2^n-PRI(*L*,*U*).

Theorem 1. *The 2^n-PRI(L,U) is feasible.*

Proof. Firstly, we prove that it is reasonable, i.e. $\sum_{i=1}^{2^n} \boldsymbol{L}_i \le \mathbf{1}$, $\sum_{i=1}^{2^n} \boldsymbol{U}_i \ge \mathbf{1}$.
We use induction on *n* instances. Let L_i^n and U_i^n be the lower bound and upper bound of the *i*-th probability interval in 2^n-PRI(*L*,*U*).

① When there is just 1 instance, let the probability interval of the instance's existence be $[l_1,u_1]$, then the probability interval of its non-existence is $[1-u_1,1-l_1]$. Now there are two possible worlds in total, and we have

$\sum_{i=1}^{2} L_i^1 = l_1 + 1 - u_1 \le 1$, $\sum_{i=1}^{2} U_i^1 = u_1 + 1 - l_1 \ge 1$. Thus the 1-PRI($L$,$U$) is reasonable.

② We now assume that when there are n-1 instances, the 2^{n-1}-PRI(L,U) is reasonable, i.e. $\sum_{i=1}^{2^{n-1}} L_i^{n-1} \le 1$, $\sum_{i=1}^{2^{n-1}} U_i^{n-1} \ge 1$.

③ When there are n instances, let the newly added instance exists with probability interval $[l_n,u_n]$, then the probability interval of its non-existence is $[1-u_n,1-l_n]$. By our assumption in ②, we have

$$\sum_{i=1}^{2^n} L_i^n = l_n * \sum_{i=1}^{2^{n-1}} L_i^{n-1} + (1-u_n) * \sum_{i=1}^{2^{n-1}} L_i^{n-1} = (l_n + 1 - u_n) * \sum_{i=1}^{2^{n-1}} L_i^{n-1} \le 1$$

$$\sum_{i=1}^{2^n} U_i^n = u_n * \sum_{i=1}^{2^{n-1}} U_i^{n-1} + (1-l_n) * \sum_{i=1}^{2^{n-1}} U_i^{n-1} = (u_n + 1 - l_n) * \sum_{i=1}^{2^{n-1}} U_i^{n-1} \ge 1$$

With ①②③, the 2^n-PRI(L,U) is reasonable.

Secondly, we also use induction to prove the feasibility. We first prove

$\sum_{i=1,i\ne j}^{2^n} L_i + U_j \le 1$, i.e. $\sum_{i=1}^{2^n} L_i^n + (U_j^n - L_j^n) \le 1$ (j=1,2,…2^n).

① When there is just 1 instance, let the probability interval of the instance's existence be $[l_1,u_1]$, then the probability interval of its non-existence is $[1-u_1,1-l_1]$. Now there are two possible worlds in total, and we have

$\sum_{i=1}^{2} L_i^1 = l_1 + 1 - u_1$, $U_1^1 - L_1^1 = u_1 - l_1$, $U_2^1 - L_2^1 = 1 - l_1 - (1-u_1) = u_1 - l_1$.

Then we get $\sum_{i=1}^{2} L_i^1 + (U_j^1 - L_j^1) = 1 \le 1$ (j=1,2), i.e. $\sum_{i=1,i\ne j}^{2} L_i + U_j \le 1$ (j=1,2).

② We assume that $\sum_{i=1,i\ne j}^{2^{n-1}} L_i + U_j \le 1$ (j=1,2,…2^{n-1}) when there are n-1 instances, i.e. $\sum_{i=1}^{2^{n-1}} L_i^{n-1} + (U_j^{n-1} - L_j^{n-1}) \le 1$ (j=1,2,…2^{n-1}).

③ When there are n instances, let the newly added instance exists with probability interval $[l_n,u_n]$, then the probability interval of its non-existence is $[1-u_n,1-l_n]$. We have

$$\sum_{i=1}^{2^n} L_i^n + (U_j^n - L_j^n) = l_n * \sum_{i=1}^{2^{n-1}} L_i^{n-1} + (1-u_n) * \sum_{i=1}^{2^{n-1}} L_i^{n-1} + (U_j^n - L_j^n) \quad (j=1,2,\ldots 2^{n-1})$$

We consider the two cases in the above formula respectively:

(I) When the n-th instance exists, $U_j^n - L_j^n = U_{j'}^{n-1} * u_n - L_{j'}^{n-1} * l_n$ (j'=1,2,…2^{n-1}).

(II) When the n-th instance does not exist, $U_j^n - L_j^n = U_{j'}^{n-1} * (1-l_n) - L_{j'}^{n-1} * (1-u_n)$ (j'=1,2,…2^{n-1}).

We consider case (I) firstly,

$$\sum_{i=1}^{2^n} L_i^n + (U_j^n - L_j^n) = l_n * \sum_{i=1}^{2^{n-1}} L_i^{n-1} + (1-u_n) * \sum_{i=1}^{2^{n-1}} L_i^{n-1} + U_j^{n-1} * u_n - L_j^{n-1} * l_n$$

$$= l_n * \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + (1-u_n) * \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + (1-u_n) * L_j^{n-1} + U_j^{n-1} * u_n$$

$$\leq (l_n + 1 - u_n) * \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + (1-u_n) * U_j^{n-1} + U_j^{n-1} * u_n = (l_n + 1 - u_n) * \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + U_j^{n-1} \cdot$$

since $0 \leq l_n + 1 - u_n \leq 1$, and by our assumption in ②, we have

$$(l_n + 1 - u_n) * \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + U_j^{n-1} \leq \sum_{i=1,i\neq j}^{2^{n-1}} L_i^{n-1} + U_j^{n-1} = \sum_{i=1}^{2^{n-1}} L_i^{n-1} + U_j^{n-1} - L_j^{n-1} \leq 1$$

Therefore with case (I), $\sum_{i=1}^{2^n} L_i^n + (U_j^n + L_j^n) \leq 1$, and similarly we can also prove that with case (II), $\sum_{i=1}^{2^n} L_i^n + (U_j^n - L_j^n) \leq 1 \cdot$ Therefore $\sum_{i=1,i\neq j}^{2^n} L_i + U_j \leq 1 \cdot$

Using similar methods we can prove that $\sum_{i=1,i\neq j}^{2^n} U_i + L_j \geq 1$ (j=1,2,…2^n), thus the feasibility is proved. □

4.2 Probabilistic Prevalence Co-locations

After proving that the 2^n-PRI(L,U) of 2^n possible worlds of n instances is feasible, we can utilize the method which is similar to the method in paper [14] to convert the probability interval of a possible world into a point probability.

Definition 4. *Let* $w_F \in W_F$, $p\_in(w_F) = [L_{w_F}, U_{w_F}]$. *The point probability* $p(\hat{w}_F)$ *corresponding to p_in(w_F) can be calculated by:*

$$p(\hat{w}_F) = L_{w_F} + \frac{1 - \sum(F)^-}{\sum(F)^+ - \sum(F)^-}(U_{w_F} - L_{w_F}) = \frac{1 - \sum(F)^-}{\sum(F)^+ - \sum(F)^-} U_{w_F} + \frac{\sum(F)^+ - 1}{\sum(F)^+ - \sum(F)^-} L_{w_F}$$

$$= \sum(F-) * U_{w_F} + \sum(F+) * L_{w_F} \qquad (1)$$

where $\sum(F)^- = \sum_{w_F \in W_F} L_{w_F}$, $\sum(F)^+ = \sum_{w_F \in W_F} U_{w_F}$,

$$\sum(F-) = \frac{1 - \sum(F)^-}{\sum(F)^+ - \sum(F)^-}, \sum(F+) = \frac{\sum(F)^+ - 1}{\sum(F)^+ - \sum(F)^-} \cdot$$

After getting the point probabilities of all possible worlds, we can define probabilistic prevalence co-locations.

Definition 5. *Let* $c = \{f_1, f_2, ..., f_k\} \subseteq F$ *be a k-size co-location,* $w_F \in W_F$.

(1)The participation ratio $PR_{w_F}(c, f_i)$ *of* f_i *(i=1,2,…,k) in c in* w_F *is defined as:*

$$PR_{w_F}(c, f_i) = \begin{cases} 1, & \text{if } |table\_instance_{w_F}(\{f_i\})| = 0 \\ \dfrac{|\Pi_{f_i}(table\_instance_{w_F}(c))|}{table\_instance_{w_F}(\{f_i\})}, & \text{otherwise} \end{cases}$$

where $table\_instance_{w_F}(c)$ *is the table instance of c in* w_F, Π *is the relational projection operation with duplication elimination.*

(2) The participation index $PI_{w_F}(c)$ *of c in* w_F *is defined as:*

$$PI_{w_F}(c) = \min_{i=1}^{k}(PR_{w_F}(c, f_i)).$$

Example 4. Taking Fig. 1 as an example, we consider c={A,B} and w_F={A.1,B.1,B.2,C.1}. Then $table\_instance_{w_F}(c) = \{\{A.1, B.1\}\}$, $\mathrm{PR}_{w_F}(c, \mathrm{A}) = 1$, $\mathrm{PR}_{w_F}(c, \mathrm{B}) = 0.5$, $\mathrm{PI}_{w_F}(c) = 0.5$.

Definition 6. *Let min_prev be the minimum participation index threshold,* $Q(c) = \{w_F \mid PI_{w_F}(c) \geq min\_prev, w_F \in W_F\}$. *The prevalence point probability* $\hat{P}(PI(c) \geq min\_prev)$ *of c is defined as:*

$$\hat{P}(PI(c) \geq min\_prev) = \sum_{w_F \in Q(c)} p(\hat{w}_F) = \sum(F-) * \sum_{w_F \in Q(c)} U_{w_F} + \sum(F+) * \sum_{w_F \in Q(c)} L_{w_F} \quad (2)$$

Definition 7. *Let min_prob be the minimum probability threshold. If* $\hat{P}(PI(c) \geq min\_prev) \geq min\_prob$, *c is called a probabilistic prevalent co-location with parameter (min_prev, min_prob).*

5 The Algorithms' Design

The basic steps of mining co-locations from spatially uncertain data with probability intervals are: firstly, computing the probability intervals of possible worlds and converting them to point probabilities according to Definition 3 and 4; secondly, generating a *k*-size co-location *c*, computing its participation index and prevalence point probability according to Definition 5 and 6; lastly, finding all probabilistic prevalent co-locations according to Definition 7.

In order to optimize the above computation, two lemmas are provided. Lemma 1 is used to improve the computation of $\sum(F)^-$ and $\sum(F)^+$ in formula (1); Lemma 2 is used to improve the computation of $\sum_{w_F \in Q(c)} L_{w_F}$ and $\sum_{w_F \in Q(c)} U_{w_F}$ in formula (2).

Lemma 1. *Let* l_i, u_i *be the lower/upper bound of the probability interval of the i-th instance.*

$$\sum(F)^- = \sum_{w_F \in W_F} L_{w_F} = \sum_{i=1}^{2^n} L_i^n = \prod_{j=1}^{n}(1-(u_j - l_j)),$$

$$\sum(F)^+ = \sum_{w_F \in W_F} U_{w_F} = \sum_{i=1}^{2^n} U_i^n = \prod_{j=1}^{n}(1+(u_j - l_j)).$$

Proof. $\because \sum_{i=1}^{2^n} L_i^n = (l_n + 1 - u_n) * \sum_{i=1}^{2^{n-1}} L_i^{n-1}$, $\therefore \sum_{i=1}^{2^n} L_i^n = (l_n + 1 - u_n) \cdots (l_2 + 1 - u_2) * \sum_{i=1}^{2^1} L_i^1$.

When there is just 1 instance, $\sum_{i=1}^{2^1} L_i^1 = a_1 + 1 - b_1$, so

$$\sum_{i=1}^{2^n} L_i^n = \prod_{j=1}^{n}(l_j + 1 - u_j) = \prod_{j=1}^{n}(1-(u_j - l_j)) \cdot$$

Similarly we get $\sum_{i=1}^{2^n} U_i^n = \prod_{j=1}^{n}(1+(u_j - l_j)) \cdot$ □

Lemma 2. *Let* $W(c) = \{w_c \mid PI_{w_c}(c) \geq min\_prev, w_c \in W_c\}$,

$$\sum_{w_F \in Q(c)} L_{w_F} = \sum_{w_c \in W(c)} L_{w_c} * \sum (F-c)^-, \text{ and } \sum_{w_F \in Q(c)} U_{w_F} = \sum_{w_c \in W(c)} U_{w_c} * \sum (F-c)^+ .$$

where W_c is the set of possible worlds generated by all instances in c. F-c is the subset of F in which features in c is moved.

Proof. Let $W_{F\text{-}c}$ be the set of possible worlds generated by all instances in *F-c*. $S_{w_c} = w_c + W_{F-c}$, which is a set of possible worlds obtained by adding the possible worlds in $W_{F\text{-}c}$ into w_c, so we have $\sum_{w \in S_{w_c}} L_w = L_{w_c} * \sum (F-c)^-$, and $\sum_{w \in S_{w_c}} U_w = U_{w_c} * \sum (F-c)^+ .$

We can see that $\bigcup_{w_c \in W(c)} (w_c + W_{F-c}) = \bigcup_{w_c \in W(c)} (S_{w_c}) = Q(c)$, Therefore

$$\sum_{w_F \in Q(c)} L_{w_F} = \sum_{w_c \in W(c)} \left(\sum_{w \in S_{w_c}} L_w \right) = \sum_{w_c \in W(c)} L_{w_c} * \sum (F-c)^-$$

$$\sum_{w_F \in Q(c)} U_{w_F} = \sum_{w_c \in W(c)} \left(\sum_{w \in S_{w_c}} U_w \right) = \sum_{w_c \in W(c)} U_{w_c} * \sum (F-c)^+$$

□

Now, we give Algorithm 1 using Lemma 1 and Lemma 2:

Algorithm 1.

Input: A spatially uncertain data set with probability intervals, the minimum participation index threshold *min_prev,* the minimum probability threshold *min_prob*

Output: All probabilistic prevalent co-locations *PPC*

Steps:

① Calculate $p\_in(w_F)$ ($w_F \in W_F$), $\sum(F-)$ and $\sum(F+)$; //by Definition 3,4 and Lemma 1

② Generate the set of all 2-size co-locations C_2 and let k=2;

③ While $C_k \neq \Phi$ Do

④ For each $c \in C_k$

⑤ { calculate $\sum_{w_F \in Q(c)} L_{w_F}$ and $\sum_{w_F \in Q(c)} U_{w_F}$; //by Lemma 2, Definition 5 and Lemma 1

⑥ calculate $P(PI(c) \geq \hat{} min\_prev)$; //by Definition 6

⑦ If $P(PI(c) \geq \hat{} min\_prev) \geq min\_prob$

⑧ $PPC \leftarrow PPC \cup \{c\}$; }

⑨ Generate the set of all k+1-size co-locations C_{k+1} and let k=k+1;

⑩ Endwhile.

When calculating the prevalence of a co-location in Algorithm 1, the calculations only happen in possible worlds where the features of the co-location exist. But the downward closure property of prevalent co-locations is not satisfied under this case. The algorithm becomes time-consuming. Fortunately, when we consider possible worlds consisted by all instances of all the features, this important property is satisfied. It is proven in Lemma 3. **Algorithm 2** mentioned in Section 6 Experimental Analysis is an improved algorithm of Algorithm 1 based on Lemma 3.

Lemma 3. $P(PI(c) \geq \hat{} min\_prev)$ *decreases monotonously as the size of c increases.*

Proof. Consider k-size co-location $c_k=\{f_1,f_2,\ldots,f_k\}$ and k+1-size co-location $c_{k+1}=c_k \cup f_{k+1}$. We have $\mathrm{P}(\mathrm{PI}(c_k) \geq \hat{} min\_prev) = \sum(F-) * \sum_{w_F \in Q(c_k)} U_{w_F} + \sum(F+) * \sum_{w_F \in Q(c_k)} L_{w_F}$

$$\mathrm{P}(\mathrm{PI}(c_{k+1}) \geq \hat{} min\_prev) = \sum(F-) * \sum_{w_F \in Q(c_{k+1})} U_{w_F} + \sum(F+) * \sum_{w_F \in Q(c_{k+1})} L_{w_F}$$

$$\because \mathrm{PI}_{w_F}(c_k) \geq \mathrm{PI}_{w_F}(c_{k+1}),\ \therefore Q(c_k) \supseteq Q(c_{k+1}),$$

$$\therefore \sum_{w_F \in Q(c_k)} U_{w_F} \geq \sum_{w_F \in Q(c_{k+1})} U_{w_F} \text{ and } \sum_{w_F \in Q(c_k)} L_{w_F} \geq \sum_{w_F \in Q(c_{k+1})} L_{w_F}$$

$$\therefore \mathrm{P}(\mathrm{PI}(c_k) \geq \hat{} min\_prev) \geq \mathrm{P}(\mathrm{PI}(c_{k+1}) \geq \hat{} min\_prev)$$ □

6 Experimental Analysis

This section verifies the effectiveness and usefulness of the proposed algorithms with synthetic and real data sets through experiments. The algorithms are implemented by Java. The experimental environment is: CPU: Intel Pentium(R) D 3.00GHz; RAM: 1GB; operating system: Microsoft Windows XP.

6.1 Experiments with Synthetic Data Sets

(1) Influence of the Distance Threshold, the Participation Index Threshold and Probability Threshold

Experimental results in Fig. 2(a) show the influence of the distance threshold d on the algorithms' efficiency. From Fig. 2(a) we can see that the execution time of both algorithms increase as d increases because a larger d indicates more row instances in the possible world and therefore the computation of participation indexes costs a longer time. For Algorithm 2, if the distance threshold is small, we can exclude the non-prevalent co-locations by Lemma 3, and its execution time is shorter than Algorithm 1. But as d increases, all the smaller-size co-locations become prevalent co-locations, so the larger-size co-locations will need computation, and then the execution time of Algorithm 2 is not significantly different from that of Algorithm 1.

The results in Fig. 2(b) illustrate the influence of the participation index threshold *min_prev* on the efficiency of the algorithms. From Fig. 2(b) we can see that as

min_prev increases, the execution time of Algorithm 1 decreases. This is because in our programming design, considering that the participation index is the minimum value of the participation ratios, when we find that the participation ratio of a feature of the co-location is smaller than *min_prev*, we will not continue to calculate the participation ratios of other features. In this way the increase of *min_prev* will accelerate the computation of the co-locations' participation indexes. For Algorithm 2, as *min_prev* gets larger, some prevalent possible worlds will become non-prevalent, and this will decrease the prevalence point probabilities of the co-locations and make larger-size co-locations to be cut off. Therefore the execution time of Algorithm 2 gets shorter and better than that of Algorithm 1.

The results in Fig. 2(c) present the influence of the probability threshold *min_prob* on the efficiency of the algorithms. From Fig. 2(c) we can see that, for Algorithm 1, since it lacks the cutting-off method, the number of possible worlds is the factor affecting the algorithmic execution time. While the increase of *min_prob* does not change the number of possible worlds, the execution time of Algorithm 1 does not change much. For Algorithm 2, as *min_prob* increases, the smaller-size co-locations may become non-prevalent, leading the larger-size co-locations to be cut off and the algorithmic execution time to decrease.

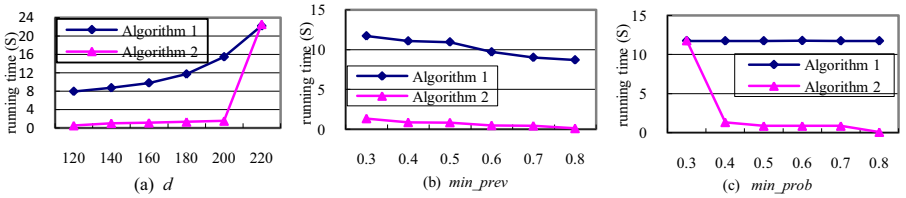


Fig. 2. The influence of threshold *d*, *min_prev* and *min_prob*

(2) The Influence of the Number of Instances and the Instances' Probability Intervals

The experimental results in Fig. 3 show the influence of the number of instances on Algorithm 2. In the experiments, take *min_prev*=0.5, *min_prob*=0.5, *d*=140, |*F*|=8.

Fig. 4 shows the influence of the instances' probability intervals on Algorithm 2. Since the co-locations' prevalence point probabilities are influenced by both the probability intervals and the distance thresholds, the change of two factors may prevent the larger-size co-locations from being cut off, and thus the algorithmic execution time grows gradually.

Fig. 5 presents the influence of the instances' probability intervals on the co-locations' prevalence point probabilities. The original data set in the figure refers to the data set with unchanged probability intervals. In the experiment, we make the lower bounds of the instances' probability intervals increase by 30%. In order to let the increased lower bounds do not exceed the upper bounds, for every $[L_{is},U_{is}]$, L_{is} is increased to be $L_{is}+(U_{is}-L_{is})*30\%$. Similarly, U_{is} is decreased to be $U_{is}-(U_{is}-L_{is})*30\%$. U_{is} is increased to be $U_{is}+(1-U_{is})*30\%$. L_{is} is decreased to be $L_{is}-L_{is}*30\%$.

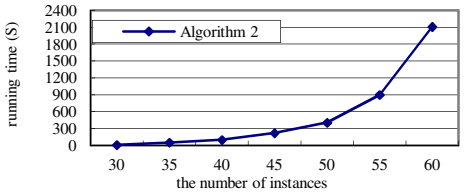


Fig. 3. The influence of the number of instances on Algorithm 2

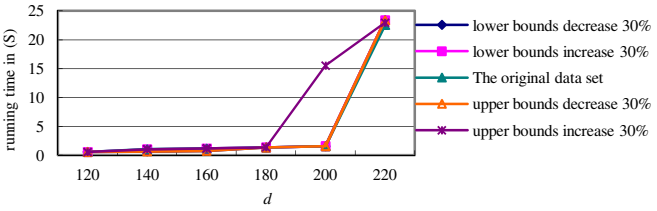


Fig. 4. The influence of the instances' probability intervals on Algorithm 2

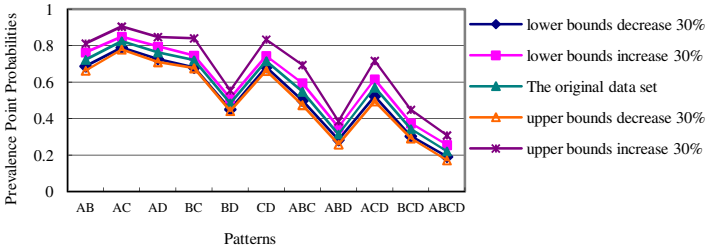


Fig. 5. The influence of the instances' probability intervals on the co-locations' prevalence point probabilities

6.2 Experiments with Real Data Sets

In this part we choose the vegetation data set of the "Three Parallel Rivers of Yunnan Protected Areas" for experiments, and analyze the influence of different parameters in the real data set on mining results. We also examine how the mining results are affected by uncertainty of the data set.

Fig. 6 gives the distribution of the data in two-dimensional space, where the X and Y coordinates represent the instances' locations relative to 0 degree longitude and latitude lines respectively in meters. Every spatial instance is randomly assigned a probability interval contained in (0,1).

In the experiment, we let the distance threshold d=1000m, the participation index threshold *min_prev*=0.35, and probability threshold *min_prob*=0.4. The mining results are shown in Table 1.

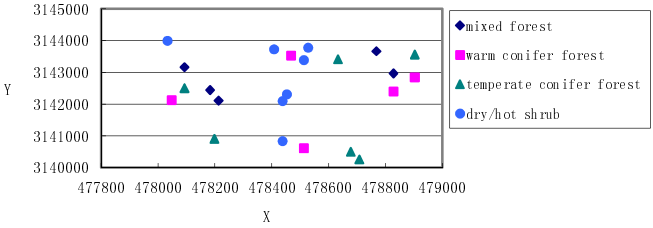


Fig. 6. Distribution of the Vegetation Data Set

Table 1. Mining Results on the Vegetation Data Set

Size	Parameter (*min_prev*,*min_prob*)=(0.35,0.4)
2	{mixed forest, warm conifer forest} { mixed forest, dry/hot shrub} {warm conifer forest, temperate conifer forest} {warm conifer forest, dry/hot shrub} {temperate conifer forest, dry/hot shrub}
3	{ mixed forest, warm conifer forest, dry/hot shrub} { warm conifer forest, temperate conifer forest, dry/hot shrub}
4	none

From Table 1 we can see that, the three vegetation warm conifer forest, temperate conifer forest, and dry/hot shrub constitute a 3-size probabilistic prevalent co-location with parameter (0.35,0.4). The parameter (0.35,0.4) means that when there is uncertainty in the plant instances' existence, the probability of this pattern being a prevalent pattern with minimum participation index threshold 0.35 is at least 40%.

Different participation index thresholds and probability thresholds will lead to different mining results. Again we use the vegetation data set in Figure 6, with distance threshold d=1000m, keep the probability intervals of the spatial instances unchanged, and adjust the participation index threshold and probability threshold respectively to compare the mining results with various parameters.

Data in Table 2 and Table 3 illustrates the mining results on the vegetation data set with different probability thresholds and different participation index thresholds, respectively. We find that both varying probability thresholds and varying participation index thresholds will give different mining results.

Table 2. Mining Results on the Vegetation Data Set with Different values of *min_prob*

(*min_prev*,*min_prob*)= (0.35,0.5)	(*min_prev*,*min_prob*)= (0.35,0.6)
{ mixed forest, warm conifer forest} { mixed forest, dry/hot shrub} {warm conifer forest, temperate conifer forest} {warm conifer forest, dry/hot shrub} {temperate conifer forest, dry/hot shrub} {mixed forest, warm conifer forest, dry/hot shrub}	{mixed forest, warm conifer forest } {mixed forest, dry/hot shrub} {warm conifer forest, temperate conifer forest} {warm conifer forest, dry/hot shrub} {temperate conifer forest, dry/hot shrub}

Table 3. Mining Results on the Vegetation Data Set with Different Values of *min_prev*

(*min_prev*,*min_prob*)=(0.45,0.4)	(*min_prev*,*min_prob*)= (0.55,0.4)
{mixed forest, warm conifer forest} {mixed forest, dry/hot shrub} {warm conifer forest, temperate conifer forest} {warm conifer forest, dry/hot shrub} {temperate conifer forest, dry/hot shrub} {mixed forest, warm conifers forest, dry/hot shrub}	{mixed forest, warm conifer forest } {mixed forest, dry/hot shrub} {warm conifer forest, dry/hot shrub} {temperate conifer forest, dry/hot shrub}

If we mine the vegetation data set without considering the uncertainty of the data set, with the same participation index threshold 0.35, distance threshold 1000m, we will get 11 prevalent co-location patterns. That means all patterns are prevalent, which is significantly different from the results in Table 1. Observe that the algorithm for mining uncertain data sets considers every possible case of the patterns' instances to get the possibility of the pattern being prevalent, and then the algorithm decides whether to mine this pattern based on the possibility of being prevalent. However, the algorithm for mining certain data sets only considers the case with all the patterns' instances existing and then decides whether this pattern is prevalent, which is not reasonable.

Thus, when there is uncertainty with the instances' existence, the results of the algorithm for mining certain data sets are not reasonable, and the algorithm for mining uncertain data sets will provide us with more satisfactory results.

7 Conclusions

This paper discusses the problem of mining co-locations from spatially uncertain data with probability intervals. Firstly, the probability intervals of all possible worlds are defined, and are proved to be feasible. Secondly, the probability intervals of possible worlds are transformed to point probabilities. The prevalence point probability and the probabilistic prevalent co-location are defined. Based on two lemmas for optimizing computation, Algorithm 1 is presented. To raise Algorithm 1's efficiency, the closure property of the prevalence point probability is proved. Based on the property, Algorithm 2 is proposed. Lastly, by experiments with synthetic and real data sets, the effectiveness and usefulness of the proposed algorithms are verified.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (No. 61063008, No. 61272126, No. 61262069), the Science Foundation of Yunnan Province (No. 2010CD025) and the Research Foundation of the Educational Department of Yunnan Province (No. 2012C103).

References

1. Wang, L., Zhou, L., Chen, H., et al.: The principle and applications of data warehouses and data mining, 2nd edn. Science press, Beijing (2009)
2. Klir, G.J., Watson, T.J.: Uncertainty and information measures for imprecise probabilities: an overview. In: The First International Symposium on Imprecise Probabilities and Their Applications (ISIPTA 1999), Ghent, Belgium, pp. 234–240 (1999)
3. Huang, Y., Shekhar, S., Xiong, H.: Discovering co-location patterns from spatial data sets: a general approach. IEEE Transactions on Knowledge and Data Engineering (TKDE) 16(12), 1472–1485 (2004)
4. Yoo, J.S., Shekhar, S.: A join-less approach for co-location pattern mining: a summary of results. IEEE Transactions on Knowledge and Data Engineering (TKDE) 18(10), 1323–1337 (2006)
5. Wang, L., Bao, Y., Lu, J., Yip, J.: A new join-less approach for co-location pattern mining. In: The 8th IEEE International Conference on Computer and Information Technology (CIT 2008), pp. 197–202. IEEE Press, New York (2008)
6. Wang, L., Zhou, L., Lu, J., Yip, J.: An order-clique-based approach for mining maximal co-locations. Information Sciences 179(19), 3370–3382 (2009)
7. Ouyang, Z., Wang, L., Chen, H.: Mining spatial co-location patterns for fuzzy objects. Chinese Journal of Computers 34(10), 1947–1955 (2011)
8. Huang, Y., Pei, J., Xiong, H.: Mining co-location patterns with rare events from spatial data sets. Geoinformatica 10(3), 239–260 (2006)
9. Feng, L., Wang, L., Gao, S.: A new approach of mining co-location patterns in spatial datasets with rare features. Journal of Nanjing University (Natural Sciences) 48(1), 99–107 (2012)
10. Lu, Y., Wang, L., Zhang, X.: Mining frequent co-location patterns from uncertain data. Journal of Frontiers of Computer Science and Technology 3(6), 656–664 (2009)
11. Lu, Y., Wang, L., Chen, H., et al.: Spatial co-location patterns mining over uncertain data based on possible worlds. Journal of Computer Research and Development, 47 (suppl.), 215–221 (2010)
12. Wang, L., Wu, P., Chen, H.: Finding probabilistic prevalent co-locations in spatially uncertain data sets. IEEE Transactions on Knowledge and Data Engineering (TKDE) 25(4), 790–804 (2013)
13. Abellan, J., Moral, S.: Maximum of entropy for credal sets. International Journal of Uncertainty, Fuzziness and Knowledge Based Systems 11(5), 587–597 (2003)
14. He, D., Zhou, R.: Study on methods of decision-making under interval probability. Journal of Systems and Management 19(2), 210–214 (2010)

A Hybrid Approach for Privacy Preservation in Location Based Queries

Zhengang Wu, Liangwen Yu, Jiawei Zhu, Huiping Sun*, Zhi Guan, and Zhong Chen

Institute of Software, EECS, Peking University, Beijing, China
MoE Key Lab of High Confidence Software Technologies (PKU)
MoE Key Lab of Network and Software Security Assurance (PKU)
{wuzg,yulw,zhujw,sunhp,guanzhi,chen}@infosec.pku.edu.cn

Abstract. With rapidly popular location-aware applications, location privacy becomes an emerging issue. This paper studies how to protect the two-fold privacy for both client-side and server-side in location-based queries. This technique is a significant component in privacy-friendly Location Based Services (LBS). Participants protect their own privacy. The LBS server protects against excessive disclose of location records in its Points of Interest (POIs) database while the mobile user protects his exact location by the cloaking technique. The proposed hybrid approach can achieve the challenging goal. Our solution integrates the cloaking technique with a cryptographic protocol, Private Set Intersection (PSI). In addition, this solution is secure in malicious model and also practical.

Keywords: Location privacy, Location Based Services, privacy-preserving protocols, Private Set Intersection, Homomorphic Encryption.

1 Introduction

Popular mobile applications have location-aware capability with the rapid de-velopment of mobile devices with built-in positioning modules. The LBS server holds a database of Points of Interest (POIs). Users can obtain POIs by sending a location-based query. The queries involve extensive applications. e.g. A user can receive personal recommendation information relevant to his location in a location-aware recommender system. However, untrusted participants may learn more sensitive information of the victim by collecting his location information. Thus protection for location privacy becomes an emerging technology.

There are extensive approaches for location privacy preservation. They fall into two major types, cloaking and cryptography. First, the cloaking is succinct and efficient for large data. However, it usually achieves a limited privacy guarantee. Second, the cryptography based approach can provide a stronger privacy guarantee but its computations are extremely expensive. Therefore a hybrid approach is necessary in a practical setting.

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 315–326, 2013.

Contributions. This paper proposes a novel hybrid approach to guarantee the client-side location privacy and the server-side content privacy in location-based queries. Only a few existing schemes [1] can achieve the strong privacy while other existing schemes only protect client-side privacy. To the best of our knowledge, our scheme is the first hybrid approach based on Private Set Intersection (PSI) for this goal.

1. Our framework focuses on a range query anchored by an exact location and integrates the query in the k-anonymous cloaking region and the PSI protocol using the grid-based partition. This solution achieves the 2-fold privacy protection that involves the location privacy of the mobile user and the content privacy of the LBS server.
2. We design the PSI protocol secure in malicious model. The LBS server can limit the size of the near range during the PSI protocol's execution. And the PSI protocol is high-efficient due to linear complexity with the size of the POI candidate results for a fixed size of the near range in the experiment.

Outline. The rest of the article is structured as follow. We summarize related works on the topic in Section 2. Preliminary is in its next section. The proposed solution is described in Section 4. Security and privacy is analyzed in Section 5. And Section 6 discusses its efficiency. In the last section we sum the solution up.

2 Related Works

Using private location-based queries, the user can query information about his location in a privacy-preserving manner. Location privacy protection is an emerging technology in several fields [2][3][4][5]. Thus private location-based queries involve a family of location privacy problems that lead to various concerns. The nearest neighbor search problem [6] concerns how to query POIs in a spatial database. e.g. A user queries his nearest hospitals. The nearby friend problem [7] is that a user queries his nearest friends privately in location-aware SNS. Friends are POIs in this scenario. Private proximity test can privately check whether the distance of two locations exceeds a threshold. Private location-based queries involve a wealth of methods which roughly fall into two catalogs.

Cryptography Based Approaches. There are several cryptographic schemes to achieve a private location-based query [1]. Private Information Retrieve (PIR) protocol is a popular building block for strong location privacy. Papadopoulos et al. [6] use secure hardware-aided PIR [8]. Ghinita et al.'s scheme [9] is a two-stage protocol that involves a private-preserving protocol for point-rectangle enclosure which is based on the Paillier Homomorphic Encryption [10] and a PIR protocol. Paulet et al.[1] improve Ghinita et al.'s scheme for performance and integrate an Obvious Transfer (OT) Protocol and a PIR protocol. In addition, zhong et al.[7] construct privacy-preserving protocols for testing location proximity of two users for the nearby friend problem. Their protocols depend on distance computation and the Paillier cryptosystem [10].

Cloaking Based Approaches. The approaches extend the exact location to a coarse-grained region in the spatial domain. i.e. the coarse-grained region covers the exact

location. Cloaking includes abundant research works because of its high efficiency. The Spatial K-anonymity techniques [11][12][13] employ the k-anonymity model for protecting location privacy. K-anonymity [14] requires that a record is anonymous or indistinguishable if at least its same k records appear in the data release. A trusted third party usually adds fake data or dummies to hide actual locations. The amount of the noise must be pondered carefully since too many dummies reduce Quality of Service (QoS). The approaches fail to guard against access pattern attack [15] since the LBS server can learn partial location information of users.

3 Preliminary

Differential Privacy

Definition 1. *The mechanism M maintains ε−Differential Privacy, iff it satisfies* $\Pr(\mathrm{M(D)}) \leq \Pr(\mathrm{M}(D_1)) \cdot e^{\varepsilon}$ *where D and* D_1 *are two neighboring databases that have a distinct row at most.*

Dwork[16][17] proposed Differential Privacy (DP) in 2006. Laplace mechanism [17] is a common DP algorithm (Equation 1) by adding random noise to protect counting query privacy. Histogram involves counting queries. Thus DP constructs a histogram in a privacy-friendly manner [18] [19].

$$\mathrm{M(x)} = \mathrm{q(x)} + \mathrm{Lap}(1/\varepsilon) \tag{1}$$

where q(x) is the query function for counting and the Laplace function Lap(1/ε) is the destiny of Laplace distribution that indicates the noise's amount.

Paillier Homomorphic Encryption

The underlying encryption for our scheme is additive Homomorphic Encryption (HE) over the integers. The Paillier cryptosystem, (Keygen,Enc,Dec), is a public-key probabilistic encryption scheme. Keygen generates a pair of the public key pk and the private key sk by a RSA modulus N (the product of two large primes). Enc is the encryption function form the additive group Z_N to the multiplicative group $\mathrm{Z}^*_{N^2}$ and Dec is the corresponding decryption function. Paillier is semantically secure against Chosen Plaintext Attacks (CPA).

Paillier holds two homomorphic properties based on modular addition. The two following equations are respectively the homomorphic addition of two messages and the homomorphic multiplication by a constant c:

$$\mathrm{Enc}(\mathrm{x}_1 + \mathrm{x}_2) = \mathrm{Enc}(\mathrm{x}_1) \cdot \mathrm{Enc}(\mathrm{x}_2)$$

$$\mathrm{Enc}(\mathrm{c} \cdot \mathrm{x}) = \mathrm{Enc}\left(\sum_{i=1}^{c} x\right) = \prod_{i=1}^{c} \mathrm{Enc(x)} = (\mathrm{Enc(x)})^c$$

Private Set Intersection

Definition 2. *Private Set Intersection is a privacy-preserving protocol for set intersection. Two participants, Alice and Bob, hold their own input sets, A and B respectively. It maintains three conditions as follow:*

1. The correctness: Alice obtains the intersection, $A \cap B$.
2. Alice's privacy: Bob fails to learn Alice's input data A.
3. Bob's privacy: Alice fails to learn Bob's additional data $B - A$.

Private Set Intersection (PSI) can compute the intersection in a privacy-preserving manner. PSI is also a cryptographic primitive to build other privacy-preserving protocols. Freedman et al. [20] proposed the first PSI protocol than depends on Obviously Polynomial Evaluation and Paillier Homomorphic Encryption.

4 Our Solution

Location-based applications need an enhanced privacy protection for both client-side and server-side. e.g. Shopular(www.shopularapp.com), a location-aware coupons issue service, can push related electronic coupons when a mobile user arrives a shop. Obviously, besides the user's location privacy, this application needs to protect its location-related coupons as a valuable asset. Otherwise rivals easily obtain excessive e-coupons by camouflaging a user. In an express delivery service scenario, a mobile customer can query his nearby couriers in the LBS server.

In these scenarios, a query issuer requests a range query anchored by his exact location. Obviously mobile users and LBS servers have fairly their own privacy requirements. First, a mobile user preserves his exact location. Second, a LBS server maintains content privacy to safeguard against excessive disclose of locations of POIs. The LBS server provides POI query services in a limited manner because its POI database is a vital fee-related asset. Therefore we define the Location-based Range Query with Strong Privacy by reference to existing works [1][6] as follow:

Definition 3. *Location-based Range Query with Strong Privacy is a privacy-preserving query processing. It involves two participants, the mobile user and the LBS server. The mobile user queries location-based data in his near range to the LBS server. It maintains three conditions as follow:*

1. The correctness: the mobile user obtains his desired data.
2. The mobile user's location privacy: his exact location is not leaked.
3. The LBS server's content privacy: locations of his additional POIs are not leaked.

4.1 Solution Summary

This scheme includes two major subroutines, a query process in a cloaking region and a private set intersection protocol, as shown in Figure 1.

In the first subroutine, a mobile user hides his exact location in a customizable Cloaking Region. A LBS server can obtain candidate results according to Cloaking Region and non-sensitive information from the user.

The second subroutine implements a private set intersection protocol in a two-party setting. The user's near range and the server's candidate results are two input sets for intersection. The user ultimately obtains his nearby POIs.

In addition, we need a conversion layer between the above two subroutines. Data of both the user and the server change into an acceptable form for the PSI protocol. Simply put, these data become positive integers.

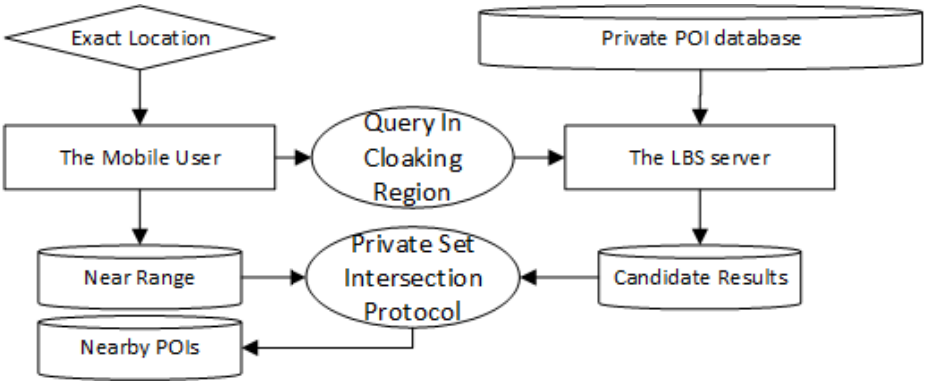


Fig. 1. Solution Summary

4.2 Computations for The Cloaking Region

A mobile user generates a cloaking region which contains the queried range anchored by his exact location. The cloaking region hides the exact location and indicates a coarse location. i.e. The coarse location suppresses the exact one sensitive. The LBS server holds a POI database. The user sends nonsensitive properties and keywords which include the coarse location to the server. Note that the queried range is a small part of the Cloaking Region.

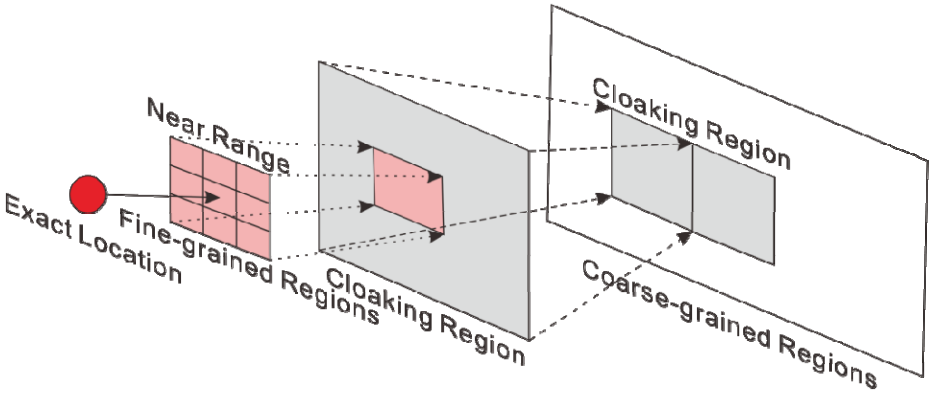


Fig. 2. The Cloaking Region and the Hierarchical Structure

GIS usually manages spatial data and geographical map hierarchically with an adjustable granularity. We can divide the whole spatial data into a two-level structure, coarse-grained regions and fine-grained regions like Figure 2 where the mobile user's near range is generalized to a cloaking region. Note that for convenience a region can be express as a set of little grids. Grids are the smallest units whose size depends on the accuracy of positioning modules.

According to Figure 1, a query in the cloaking region involves computations in both sides. Algorithm 1 and Algorithm 2 depend on k-anonymity and differential privacy respectively. Let Count() be a counting function.

The Mobile User's Computation. In this stage, the mobile user creates a cloaking region to hide his exact location. K-anonymity requires that the user number in the cloaking region exceeds k. If the user number in the local region fails to satisfy k-anonymity, Algorithm 1 constructs the cloaking region by putting together the local region and other coarse-grained regions. In non-trivial situations, Algorithm 1 has a constant time complexity. Note that Count() may need a Trusted Third Party (TTP) to log the statistics but we can estimate the Count() value without a TTP by background knowledge since the cloaking region is not precise.

Algorithm 1: Cloaking-k: Creating The Cloaking Region for Users

Input: The parameter k for k-anonymity;
The current location of the mobile user;
The size limit of the Cloaking Region;
Output: The Cloaking Region CR

```
The user builds his Near Range (NR) anchored by his current exact location.;
CR ⟵ the coarse-grained region which includes NR;
n ⟵ Count(Users in CR)/* It may be an estimated value*/;
while n < k do
    Check whether the current CR satisfies k-anonymity;
    Choosing a coarse-grained region CR_1 randomly;
    CR ⟵ CR ⋃ CR_1;
    n ⟵ n+Count(Users in CR_1);
return CR;
```

The LBS Server's Computation. In this stage, the LBS server needs to protect against excessive disclose of the size of POIs in the received cloaking region. Because a user can draw a histogram according to the size of POIs in the received cloaking region no matter what PSI. The histogram leaks statistical distribution following distinct regions. And PSI fails to protect the size of the POI set. Algorithm 2 is based on Differential Privacy. It adds a Laplace noise over the actual data. It has also a constant time complexity and the dummies will be blinded through the PSI protocol.

Algorithm 2: NoisyQuery: Retrieving Candidate Results for Servers

Input: The parameter ε for ε-differential privacy;
The Cloaking Region CR from the mobile users;
Output: The candidate results relevant to the Cloaking Region.

```
Querying actual locations (a set L) of POIs in the Cloaking Region;
n ⟵ Count(actual locations of the POIs);
Computing the amount Lap(1/ε) of the noise;
Generating dummies (a set D) whose number is the noise's ceiling, ⌈Lap(1/ε)⌉;
Candidate Results is the union L ⋃ D of actual locations and dummies;
return Candidate Results;
```

4.3 Transformation: Preparing for Intersection

We transfer this location-based query problem into the set intersection problem as Fig.3 shows. A geographical map is overlaid with a grid network. A unique integer, Grid ID labels a little grid. Each location falls into the corresponding grid. Some close locations may become a same Grid ID due to precision. The user U's near range can be expressed as a set of Grid IDs like Subfigure(A) where a cross-shaped symbol means a Grid ID in the near range. In a similar way, the LBS server holds also a Grid ID set of locations of POIs (hearts) as Subfigure(B). Thus it is straightforward to obtain the nearby POIs, by the intersection of two Grid ID sets held by the user and the server respectively.

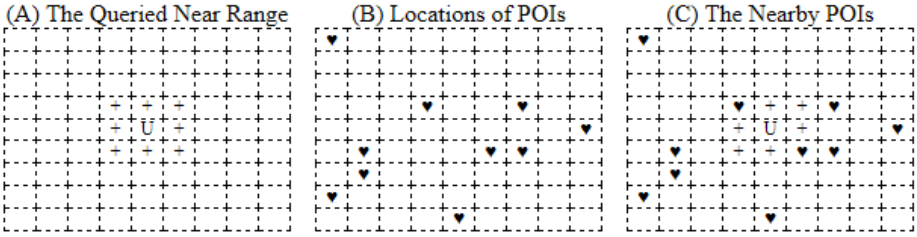


Fig. 3. Computation to Obtain Nearby POIs

The transformation way from locations (coordinates or other similar things) to Grid IDs is flexible. The representation based on the Grid ID set is independent of the shape of the spatial region but obviously the mobile and the LBS server employs the same transformation to ensure the semantic consistency.

4.4 Private Set Intersection for POIs

We propose a PSI variant based on Freedman's PSI [20] to protect the enhanced privacy for both the user and the server in the process of calculating intersection. Our PSI is secure in the presence of malicious users. Greed users cannot obtain excessive POIs. First, users fail to send an excessive near range without the server's permission. The PSI supports a security policy to effectively check the number of the received ciphertexts. Second, the server cannot compute a malicious polynomial whose coefficients are all-zero. The attack using the malicious polynomial, $F(x) \equiv 0$, are discussed in [20][21][22]. We propose an effective method to achieve immunity against the attack.

Algorithm 3 demonstrates our PSI. The mobile user holds a location set A on behalf of his near range. Similarly, the LBS server holds another location set B of POIs in the queried cloaking region. Elements of Both A and B are Grid IDs each of which labels a small piece of location.

Let $|A| = s$ and in the Paillier cryptosystem, Keygen(), Enc() and Dec() are the secret-key generation, the encryption and the decryption respectively.

Algorithm 3: PSI-Location:Private Set Intersection for Locations of POIs

Input: The mobile user holds the NR set A of size s, $\{a_1, a_2, \cdots, a_s\}$.
The LBS server holds the POI set B of size n, $\{b_1, b_2, \cdots, b_n\}$.
The Near Range is not allowed to exceed the maximum size L.

Output: Alice obtains its nearby POIs which belong to the intersection $A \bigcap B$.

1: (Setup phase) Client:$(pk, sk) \leftarrow Keygen(1^k)$ and Client $\longrightarrow$ Server:pk
2: Client encodes A into a polynomial $F(x) = x^s + \sum_{j=0}^{s-1} c_j \cdot x^j$.
Client $\longrightarrow$ Server:$Enc(c_0), Enc(c_1), \cdots, Enc(c_{s-1})$
3: Server receives $Enc(c_0), Enc(c_1), \cdots, Enc(c_{s-1})$.
Server breaks the request if $s > L$, otherwise continues the protocol.
Server computes $Enc(r(F(x)) + x)$ for each $x \in B$ where r is a random integer.
Server obtains the encrypted set $Enc(B') = \{Enc(r(F(x)) + x) | x \in B\}$
Server permutes randomly elements of $Enc(B')$ and sends them to Client.
4: Client decrypts the received ciphertexts $Enc(B')$ to obtains $B^{'}, Dec(Enc(B'))$.
Client computes the set intersection $A \bigcap B^{'}$ that is exactly $A \bigcap B$.

We have some vital points to explain this protocol involving the following three equations.

$$F(x) = \prod_{i \in A} (x - i) = 1 \cdot x^s + \sum_{j=0}^{s-1} c_j \cdot x^j \quad (2)$$

$$Enc(F(x)) = Enc(x^s) \cdot Enc(\sum_{j=0}^{s-1} c_j \cdot x^j) = Enc(x^s) \cdot \prod_{j=0}^{s-1} Enc(c_j \cdot x^j) \quad (3)$$

$$Enc(rF(x) + x) = (Enc(F(x)))^r \cdot Enc(x) \quad (4)$$

First of all, our PSI protects against the malicious polynomial $\mathrm{F(x)} \equiv 0$. In Step 2, the coefficient of the highest item x^s in Equation 2 is 1 invariably. So the user encrypts only the rest of the F(x) coefficients, i.e. $c_0, c_1, \cdots, c_{s-1}$. Above all, $\mathrm{F(x)} \equiv x^s$ if $c_0, c_1, \cdots, c_{s-1}$ are all-zero. Obviously, $\mathrm{rF(x)} + \mathrm{x} \equiv \mathrm{x}$ for all integers in B when $\mathrm{F(x)} \equiv 0$ and thus the whole B will be sent.

Next, the server computes $\mathrm{rF(x)} + \mathrm{x}$ inputting all values in B according to homomorphic properties in Step 3. Equation 3 and 4 show the secure computation for $\mathrm{rF(x)} + \mathrm{x}$. $\mathrm{rF(x)} + \mathrm{x} = \mathrm{x}$ since x is roots of F(x) if $\mathrm{x} \in \mathrm{A} \cap \mathrm{B}$. For another, if $\mathrm{x} \notin \mathrm{A} \cap \mathrm{B}$, $\mathrm{rF(x)} + \mathrm{x}$ is also random because of a random integer r. Thus $\mathrm{rF(x)} + \mathrm{x}$ blinds elements in $\mathrm{B} - \mathrm{A}$.

5 Security and Privacy

Our solution guarantees the two-fold privacy which satisfies privacy requirements of both the client and the server. This scheme is secure in malicious model.

The Correctness. The whole scheme's correctness is easily proofed by the PSI's correctness. Equation 4 ensures that for all $\mathrm{x} \in \mathrm{A} \cap \mathrm{B}$, $\mathrm{rF(x)} + \mathrm{x} = \mathrm{x}$ since x is the F(x)'s roots. Actually, $\mathrm{A} \cap \mathrm{B} \equiv \mathrm{A} \cap \mathrm{B}'$ although $\mathrm{B} \neq \mathrm{B}'$.

The Client's Location Privacy. Two techniques, Cloaking and PSI, protect the client's exact location. First, the exact location is generalized to the cloaking region using Algorithm 1 based on k-anonymity. Second, in our PSI, the Paillier homomorphic encryption's semantical security ensures that the LBS server learns nothing on the user's near range except its size that equals the degree of the polynomial F(x). Note that the server checks the query permission through the knowledge of the near range's size.

The Server's Content Privacy. The LBS server has to protect against excessive disclose of its POI database. DP and PSI preserve its content privacy that actual locations of POIs are sensitive.

First, our PSI protocol randomizes these POI locations in the Cloaking Region (CR) but not in the Near Range (NR). i.e. these integer values in the additional region CR-NR are meaningless through PSI. We can proof it using Equation 4 and the Paillier's homomorphic properties.

Second, the size of the actual POI sub database in the Cloaking Region is still sensitive obviously. We add the appropriate noise to the actual size by Algorithm 2 since our PSI fails to hide the actual size. Laplace mechanism satisfying differential privacy indicates the amount of the additional noisy data.

Security Against the Malicious User. Usually, semi-honest adversaries (passive attackers) only eavesdrops data. However, malicious adversaries (active attackers) can disturb the protocol's execution by elaborating input data. Our PSI protocol can protect against two malicious behaviours as follow.

1. **The Overlarge Queried Region:** The malicious user may submit dishonestly an overlarge region to obtain excessive POIs. Step 3 of Algorithm 3 protects against this attack. Because the length of the received $\mathrm{Enc}(c_j)$ sequence equals the size of the user's near range according to Equation 2 and Step 2 of Algorithm 3. The server only responses approved requests by this binding.
2. **The All-zero Polynomial:** The malicious user may submit dishonestly the degenerated polynomial [20][22],$F(x) \equiv 0$, whose coefficients are all-zero. In this way the server discloses the whole set B computing $\mathrm{Enc}(B')$ by $rF(x) + x \equiv x$. However, to prevent it, our PSI uses a succinct new technique that the server can avoid employing the encrypted coefficient $\mathrm{Enc}(c_s)$ of the highest item x^s according to Equation 3. The user fails to send $\mathrm{Enc}(c_s)$ since $c_s \equiv 1$ in Equation 2. In the worst situation, the polynomial F(x) in the server can degenerate into $F(x) \equiv x^s$ and thus the malicious user fails to receive any meaningful values through the randomness of $rx^s + x$.

6 Efficiency

Our scheme's efficiency depends on the PSI protocol that outputs the majority of both time complexity and communication cost as the following table. Let the size of the server set (candidate results) be β and the size of the client set (near range) be a.

Note that usually $a \leq \beta$ and $a < L$ is usually a small integer. The modulus N of Paillier is at least 1024 bits. The Paillier's ciphertext is twice as long as the corresponding plaintext.

Subroutine	Time Complexity		Communication Cost	
	Client	Server	Client→Server	Server→Client
Query	O(1)	O(1)	O(1)	0
PSI	$O(a+\beta)$	$O((a+1)\beta)$	Na	$2N\beta$

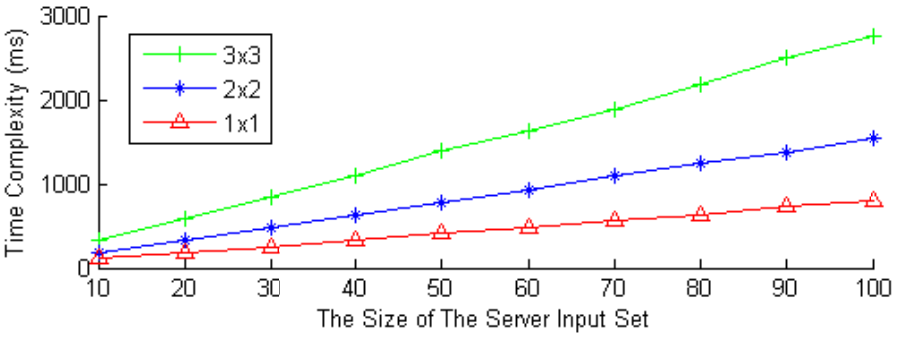


Fig. 4. Performance

Fig. 4, where the unit of time complexity is millisecond (ms), shows the performance of our PSI protocol in single thread mode. We implement it using JAVA. The execution can be within seconds in a laptop with an Intel Core i7 2.4GHz CPU and 8G RAM. It has approximately a linear complexity for a given NR size (3×3, 2×2 or 1×1) regardless of the NR's shape. However, a larger NR size leads to a more complex polynomial that increases the amount of calculation.

In this experiment, each Grid ID has 6 decimal numbers and the server of input set has a variable size because of the addition of the DP noise. We log the total time of the protocol as Fig.4 shows. The client executes the majority of encryption and all of decryption computations and the server calculates wholly the encrypted polynomial $\mathrm{Enc}(rF(x) + x)$ for all $x \in B$ over ciphertexts.

7 Conclusion

This paper discusses private location-based queries where our solution can guarantee the two-fold privacy: the mobile user needs not to leak his exact location; the LBS server protects against excessive disclose of its POI database. The proposed hybrid approach integrates the cloaking technique and the privacy-preserving protocol. We transform queries on the near range located on the cloaking region into secure computations for set intersection. Our approach is secure in malicious model to protect against the user's malicious data. In addition, it is practical and high-efficient according to discussion on its efficiency.

Acknowledgment. This work is partially supported by the HGJ National Significant Science and Technology Projects under Grant No. 2012ZX01039-004-009, Key Lab of Information Network Security, Ministry of Public Security under Grant No.C11606, the National Natural Science Foundation of China under Grant No. 61170263

References

1. Paulet, R., Kaosar, M.G., Yi, X., Bertino, E.: Privacy-preserving and content-protecting location based queries. In: ICDE, pp. 44–53 (2012)
2. Jian, Y., Chen, S., Zhang, Z., Zhang, L.: Protecting receiver-location privacy in wireless sensor networks. In: IEEE INFOCOM, pp. 1955–1963 (2007)
3. Huang, Y., Vishwanathan, R.: Privacy preserving group nearest neighbour queries in location-based services using cryptographic techniques. In: IEEE GLOBECOM, pp. 1–5 (2010)
4. Li, Y., Ren, J.: Source-location privacy through dynamic routing in wireless sensor networks. In: IEEE INFOCOM, pp. 2660–2668 (2010)
5. Pingley, A., Zhang, N., Fu, X., Choi, H.A., Subramaniam, S., Zhao, W.: Protection of query privacy for continuous location based services. In: IEEE INFOCOM, pp. 1710–1718 (2011)
6. Papadopoulos, S., Bakiras, S., Papadias, D.: Nearest neighbor search with strong location privacy. PVLDB 3(1), 619–629 (2010)
7. Zhong, G., Goldberg, I., Hengartner, U.: Louis, lester and pierre: Three protocols for location privacy. In: Borisov, N., Golle, P. (eds.) PET 2007. LNCS, vol. 4776, pp. 62–76. Springer, Heidelberg (2007)
8. Williams, P., Sion, R.: Usable pir. In: NDSS. The Internet Society (2008)
9. Ghinita, G., Kalnis, P., Kantarcioglu, M., Bertino, E.: Approximate and exact hybrid algorithms for private nearest-neighbor queries with database protection. GeoInformatica 15(4), 699–726 (2011)
10. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
11. Mokbel, M.F., Chow, C.Y., Aref, W.G.: The new casper: A privacy-aware location based database server. In: Chirkova, R., Dogac, A., Özsu, M.T., Sellis, T.K. (eds.) IEEE ICDE, pp. 1499–1500 (2007)
12. Mokbel, M.F., Chow, C.Y., Aref, W.G.: The new casper: Query processing for location services without compromising privacy. In: Dayal, U., Whang, K.Y., Lomet, D.B., Alonso, G., Lohman, G.M., Kersten, M.L., Cha, S.K., Kim, Y.K. (eds.) VLDB, pp. 763–774. ACM (2006)
13. Kalnis, P., Ghinita, G., Mouratidis, K., Papadias, D.: Preventing location-based identity inference in anonymous spatial queries. IEEE Trans. Knowl. Data Eng, 1719–1733 (2007)
14. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information (abstract). In: Mendelzon, A.O., Paredaens, J. (eds.) PODS, p. 188. ACM Press (1998)
15. Williams, P., Sion, R., Carbunar, B.: Building castles out of mud: practical access pattern privacy and correctness on untrusted storage. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM Conference on Computer and Communications Security, pp. 139–148. ACM (2008)

16. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
17. Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)
18. Xu, J., Zhang, Z., Xiao, X., Yang, Y., Yu, G.: Differentially private histogram publication. In: ICDE, pp. 32–43 (2012)
19. Hay, M., Rastogi, V., Miklau, G., Suciu, D.: Boosting the accuracy of differentially private histograms through consistency. PVLDB 3(1), 1021–1032 (2010)
20. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004)
21. Jarecki, S., Liu, X.: Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577–594. Springer, Heidelberg (2009)
22. Hazay, C., Nissim, K.: Efficient set operations in the presence of malicious adversaries. J. Cryptology 25(3), 383–433 (2012)

Dynamic Table: A Scalable Storage Structure in the Cloud*

Hanchen Su, Hongyan Li**, Xu Cheng, and Zhiqiang Liu

Key Laboratory of Machine Perception (Peking University), Ministry of Education
School of Electronics Engineering and Computer Science,
Peking University Beijing, China
{suhanchen,lihy,chengxu,liuzq}@cis.pku.edu.cn

Abstract. Big data bring us not only constantly growing data volume, dynamic and elastic storage demands, diversified data structures, but also different data features. Apart from the traditional dense data, more and more "sparse" data emerged and account for the majority of the massive data. How to adapt to the characteristics of the sparse data without losing sight of the traits of the dense data is a challenge. This paper studies how to integrate row and column data-layouts for both dense and sparse datasets in the cloud. A new NF^2 scalable storage structure named "Dynamic Table" based on the key-value storage is proposed. The formal definition of dynamic table and implemention on HDFS is also introduced.

Keywords: Massive Data, NF^2, Cloud Computing, HDFS.

1 Introduction

As information technology developing rapidly, a big data era is coming. Massive data arises in a variety of applications such as recommendation systems on social network, mobile Internet and unstructured data management. The rate of data growth is very fast: doubling every 18 months. Besides, in addition to the traditional relational data, much more unstructured data such as video, image and text are generated. According to a study of IDC, in 2012, 1800 EB digital information is generated and 80% of it is unstructured data[1] including image, audio, video, text and so on. Compared with other appliances, the cloud storage solution and computing techniques offers almost unlimited storage space and computing power together with high scalability, dynamic elasticity, in-database processing with MapReduce paradigm and pay-as-you-go service delivery pattern. However there are still many difficulties in storing and managing big data in such a multi-tenant circumstance:

* This work is supported by Natural Science Foundation of China (NSFC) under grant numbers: 60973002 and 61170003, and National Science and Technology Major Program (No.2010ZX01042-001-003-05, 2010ZX01042-002-002-02).

** Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 327–338, 2013.

1. The amount of the data sets on the cloud is very large, the schemas of it is also complicated: data sets usually have hundreds of or even thousands of attributes. This is often the case that people would like to have tuples that range over only a part of the relation schema, and the sheer number of possible attributes is very large. Besides, the schema is various and unstable: new attributes are added and old attributes may be abandoned with the evolution of the understanding of the data. In many cases, sparse data and dense data are mixed corresponding to distinct subsets of the entity's attributes. The processing method of the missing values is more complicate. Take the storage of images in Figure 1 as an example. Assume that images is stored in a traditional relational database; their basic information (Filename, Type, Size) are very dense that there are almost no null values in this attribute set. On the other hand, there are many absent values in the attribute like semantic descriptions (content and tone) and color features vectors (RGB, HSV[3] and Tamura[4]) because not all of them are exist or needed. Layout methods for dense and sparse data are often seen as two separate problems. Existing storage solutions didn't supply more than one physical storage layout for choice under a unified logical model.

id	Name	Type	Size	Content	Tone	RGB	HSV	Tamura
img1	Flower	jpg	341k	Flower	Yellow	V_1	V_2	-
img2	Art	png	502k	-	Red	-	-	V_3
img3	PKU	bmp	108k	Campus	-	-	-	-
img4	File001	jpg	311k	-	-	-	V_4	V_5

Fig. 1. A Simple Example for Image Storage in Relational Model

2. Diverse applications like frequent queries and MapReduce-based data mining, machine learning and other computations on the data sets need high space utilization rate, good query mechanism and flexible schema scalability requiring the advantages of both row-oriented and column-oriented storage, which is difficult to be implemented in the popular key-value pattern in cloud storage. Existing cloud data systems, either traditional relational model or non-relational data model, are mainly developed for a certain kind of application and are not sufficient to deal with mixed datasets efficiently.

Most of the current famous data manage systems on cloud are either using the RDMBS model which have tight constraints on the schema of the data (Hive), which will cause a big waste of storage space; or the physical organization is too "loose" (HBase, Pig, Cassandra) so that the data that are close to each other in logic layer (or say, in one column) may be stored separately, which will cause low query performances. In fact, most of the systems don't have a convenience structured query language like SQL for users.

Intrigued by such an observation, we studied how to integrate row and column data-layouts for both dense and sparse datasets in the cloud. A new NF^2 scalable storage structure named "Dynamic Table" based on the key-value storage is proposed. The columns in dynamic table are grouped and tuples may have different schema.

Besides, the schema of the table can dynamically modify to support the flexibility and scalability. The formal definition of dynamic table and the algebra are given. We also implement the dynamic table in an unstructured data management system – Muldas based on Hadoop. Users of Muldas could organize the columns of the table into different subsets and make choice of row-oriented or column-oriented storage strategy according to different applications. A structured SQL-like query language called MQL on Muldas is also supported.

This paper is organized as follows. Section 2 gives the previous related work. Section 3 gives the formal definition of dynamic table. Section 4 discusses the way of handling the absent value of dynamic table. Section 5 gives the operator definition to support the queries and Section 6 discuss the implementation of Muldas based on key-value pattern. Finally we present an experimental comparison of our model to several open source data stores in section 7.

2 Related Work

There are various methods proposed in order to handle the data sets on cloud that include large scale sparse data or have a flexible schema. Based on the logical model or physical layout, they can be divided into the following categories.

The first category tries to handle the data with the original relational data model. The main idea is to give a uniformed schema that covers all the attributes related to the data and uses "null" placeholder or use column storage (such as C-Store[5], MonetDB[6]) to ignore the absent values in the case that the tuple has no values on the specified attribute; while the former solution has a very low data utility rate and the latter one slows down the performance of queries and other computations that need many columns because of it may require many join operations. The most famous system on cloud using the relational data model is Apache Hive[6]. Hive uses the RCFile[7] applies the concept of "first horizontally-partition, then vertically-partition" as well. The main drawback of these systems that use traditional relational is that it is difficult to handle the data with a flexible schema and they need suffers from the storage space waste of null values.

The second category uses offers a new self-defined language and analyze the data which stores the data as normal files in local or cloud file system. Typical example of this category is the Pig by Yahoo. It uses a language called Pig Latin to turn the query into multiple map-reduce jobs to finish the work. However, Pig Latin is not as convenient as SQL. Pig Latin is a data flow programming language, whereas SQL is a declarative programming language. In other words, a Pig Latin program is a step-by-step set of operations on an input relation, in which each step is a single transformation. By contrast, SQL statements are a set of constraints that, taken together, define the output. In many ways, programming in Pig Latin is like working at the level of an RDBMS query planner, which figures out how to turn a declarative statement into a system of steps. Besides, Pig does not have a well-designed storage layer plan. The data are all stored in files or other format defined by user. Users of Pig have to write their own User Defined Functions (UDF) to process the specified data, which still causes a lot of work.

The third category is known as the BigTable data storage method. The most famous storage systems are Hbase[10] and Cassandra[11] using the Google BigTable[12] model which divide the columns into "column family" and store the values in key value pairs in each column family; in this way it to "decompose" a sparse table into a number of smaller and denser parts; they support adding and deleting columns and column families. However, the "column" in column family is a tag attached to the value to make a key-value pair. And the key-value pairs in one column family are stored according to the order of the timestamp the key-value inserted. So the "tuple" in BigTable is neither stored in row layout nor in column layout; the performance of querying in Hbase is very slow. Hbase does not support structured query language. Users of HBase have to use the raw API to get data out of the table.

3 Definition of Dynamic Table

In this section the concept and structure of the data model are introduced: dynamic table and dynamic tuple. A dynamic table and dynamic tuple is defined in two aspects, the schema and content; to define a dynamic table and dynamic tuple, the types and values in the dynamic table are defined first.

3.1 Types and Values of Dynamic Table

Definition 3.1. The types of dynamic table can be recursively defined as follow:
A *table type* Tt = [Ref, CG_1, CG_2, … , CG_n], where CG_i = <N_i, Cs_i> ($1 \le i \le n$) is a *column group type* and Ref is a type called *identifier type*; Ref is a type called *identifier type*: Dom(Ref) = {id_1, id_2, … , id_n} where id_i is an ID allocated by system to mark a row;

A *column group type* CG_i = <N_i, Cs_i> Where N_i is the column group name with the condition $N_i \neq N_j$ if $i \neq j$ and Cs_i = (C_{i1}, C_{i2}, … , C_{in}) ($1 \le i \le n$) where C_{ij} = <A_{ij}, B_{ij}> is a *column type*;

A *column type*: C_{ij} = <A_{ij}, B_{ij}>, where A_{ij} is the attribute name of column C_{ij} and B_{ij} is a basic type like integer, string, other common types and their combinations in most database system;

A *tuple type*: If Tt = [Ref, CG_1, CG_2, …, CG_n] is a *table* type, then TPt = [Ref, CG_i,…,CG_k] where [CG_i,…,CG_k]∈[CG_1, CG_2, … , CG_n] is a *tuple type* defined on a *table type* Tt;

Definition 3.2. The values can be defined as follow:

identifier value: If $o \in$ Dom(Ref), the o is *identifier value*;

attribute value: If v is value of basic type or v is a *identifier value*, then v is an *attribute value*; $v \in M \cup$ Dom(Ref); Where M is the domain of all basic types;

attributes group value: If CG = <N_i, (<A_1, B_1>, … ,<A_n, B_n>)> is *column group type* and v_1, v_2, … , v_n is *attribute value* with $v_i \in$ Dom(B_i); then agv = (av_1, … , av_n) is an *attributes group value*; Dom(CG)={(v_1, v_2, … , v_n)|$v_i \in$ Dom(B_i)};

tuple value: If o is *identifier value*, agv_1, agv_2,…agv_n is *attributes group value* from one table; TPt = [Ref, CG_1…CG_n] where CG_i = <N_i, Cs_i> is a *tuple type* of this table, then tpv = [o, N_1:agv_1, N_2:agv_2,…, N_n:agv_n] is a *tuple value* interpreted from TPt: Dom(TPt) = {[o, agv_1, agv_2,…, agv_n]| $o \in$ Dom(Ref) and $v_i \in$ Dom(CG_i)};

Here are some examples of types and values:

column group type: [Basicinfo: (name: string, size: integer)];

dynamic table type T: [Ref, Basic Info: (name: string, size: integer), Semantic: (content: string tone: string), Feature Vector: (RGB:integerVector, HSV:intgerVector, Tamaura: intgerVector)].

tuple type defined on T: [Ref, BasicInfo: (name: string, size: integer)];

attribute value: size:128k;

tuple value: [id1, Basic Info: (name: PKU.bmp, size:108k)] Semantic: (Content:campus,Tone: Null)];

Now we can define the dynamic table and dynamic tuple.

Definition 3.3. A dynamic table DT = <DTs, DTv> and a dynamic tuple of this table TP = <TPs, TPv> where DTs is a *table type*; TPs is a *tuple type* defined on DTs; TPv is a *tuple value* interpreted from TPs and DTv is the set of all dynamic tuple of DT. DTs and TPs are called the current schema of DT and TP. Denote AttG(TP) = TPs – {Ref} the set of attribute groups of TP; Attr(TP) = $\{C_{ij} | C_{ij} \in CG_i, CG_i \in AttG(TP)\}$ the attributes set of TP; If CG_i is a column group type $CG_i \in TPs$ and N_i is a column group name of CG_i and A_{ij} is the attribute name of column C_{ij}, then we use $TP[N_i]$ and $TP[N_i.A_{ij}]$ to present the correspondent attributes group value and attribute value of TP.

As shown in the definition, the schema of dynamic tuples in one dynamic table can be different; some groups of attributes may not exist. The schema of a dynamic table is superset of the schema of its tuples.

id	Basic Info			Semantic		Feature Vectors		
	Filename	**FileType**	**Size**	**Content**	**Tone**	**RGB**	**HSV**	**Tamura**
id1	Flower	jpg	341k	Flower	Yellow	V_1	V_2	Null
id2	Art	png	502k	Null	Red	Null	Null	V_3
id3	PKU	bmp	108k	Campus	Null	Vacant	Vacant	Vacant
id4	File001	jpg	311k	Vacant	Vacant	Null	V_4	V_5

Fig. 2. Dynamic Table for Simple Example of Figure 1

3.2 Absent Values in Dynamic Table

In dynamic table, the absent values need to be expanded since the structure of the table and the tuple can be different. The absent values in a dynamic table may have two semantics:

Null: the attribute of this tuple is included in the scheme of but the value of this attribute is unknown, or not generated.

Vacant: the attributes of this column group are not a part of the schema of this tuple: For a dynamic tuple TP = <TPs, TPv> and a column group CG_i of the dynamic table this TP is defined on, if $CG_i = <N_i, Cs_i> \notin TPs$, then $TP[N_i]$ = Vacant; and for all $C_{ij} = <A_{ij}, B_{ij}> \in Cs_i$, $TP[N_i.A_{ij}]$ = Vacant;

3.3 Schema Modification of Dynamic Table and Dynamic Tuple

In dynamic table, dynamic changes on schema are supported: column groups can be added/deleted into/from the current schema of the dynamic table, and columns can also be added/deleted into/from the current column groups. With the schema modification, flexibility and scalability of unstructured data are supported in dynamic table.

Definition 3.4. Adding/deleting columns groups into/from dynamic table or tuple
If DT = <DTs, DTv> where DTs = [Ref, CG_1, CG_2, … , CG_n],

If CGs = [CG_1', CG_2',…CG_m'] and $CG_i' \notin$ DTs then $DT^{new} = DT^{old}$ + CGs := <DTs′, DTv> where DTs′ = [Ref, CG_1, CG_2, … , CG_n, CG_1', CG_2', … , CG_m'];

If CGs = [CG_i,CG_j, … ,CG_k] where CGs ⊆ DTs then $DT^{new} = DT^{old}$ - CGs := <DTt′, DTv′> where DTt′ = [Ref, GG_1, … ,CG_n] - [CG_i,CG_j, … ,CG_k], DTv′ = {tp′| tp′ = <TPs - CGs, tp[TPs-CGs]>, tp∈DTv};

Definition 3.5. Adding/deleting column into/from column groups
If DT = <DTs, DTv> where DTs = [Ref, CG_1, CG_2, … , CG_n]

If C_x = <A_x,B_x> is *column type* CG_i is a *column group type* of DTs; ∄C_{ij} that A_{ij} = A_x, then $CG_i^{new} = CG_i^{old} + C_x$:= {C_{i1},…C_{in}, C_x}, DT^{new} = <DTs′, DTv′> where DTs′ = DTs – CG_i^{old} + CG_i^{new} and DTv′ = {tp′| tp′ = tp + C_x:Null, tp ∈ DTv }

If C_{ij} = <A_{ij}, B_{ij}>∈CG_i then $CG_i^{new} = CG_i^{old} – C_{ij}$, DTs′ = DTs – CG_i^{old} + CG_i^{new} and DTv′ = {tp′| tp′ = tp – tp[A_{ij}], tp ∈ DTv }

According to the definition:

When adding a new column groups for a dynamic table, the tuple will remain the same (the values on these new attributes are all Vacant);

When adding a new column into a column group for a dynamic table, the tuples that have value on attributes of this column group will have a new attribute and the value is Null;

When deleting a column group or a column for a dynamic table, the schema of the dynamic table and its tuples are changed; for the tuples that have on the corresponded attribute(s), the attribute(s) will be deleted.

And for a dynamic tuple:

When adding an attributes group that defined on the table but not included in the schema of the tuple, the attributes will be added into the schema of the tuple and the value for all the attributes of this column group will be Null for initialization;

When deleting an attributes group for a tuple, the attributes will be deleted from the schema of the tuple (the values of these attributes will become Vacant).

3.4 Select Operation on Dynamic Table

In order to support the queries on dynamic table, the select operators on dynamic table need to be defined. Since the schema of dynamic table and tuples can be different, to define the operators, the equation relations between types and values must be defined first:

Definition 3.7. The equation "$=_t$" and "$=_T$" between types are defined as follow:
If T is a basic type then T $=_t$ T;

column type: If $C_i = <A_i, B_i>$ and $C_i = <A_j, B_j>$ is a *column type*; $C_i =_t C_j$ iff $B_i =_t B_j$; $C_i =_T C_j$ iff $A_i = A_j \wedge B_i =_t B_j$

column group type: If $CG_i = <N_i, Cs_i>$ and $CG_j = <N_j, Cs_j>$ are *column type*, $Cs_i = (C_{i1}, C_{i2}, \ldots, C_{in})$ and $Cs_j = (C_{j1}, C_{j2}, \ldots, C_{jm})$; then $CG_i =_t CG_i$ iff n = m and $\forall x\ C_{ix} =_t C_{jx}(1 \le x \le n)$; $CG_i =_T CG_i$ iff $n = m \wedge N_i = N_j \wedge \forall x\ C_{ix} =_T C_{jx}(1 \le x \le n)$;

tuple type: If tpv_i = [Ref, $CG_{i1}, \ldots, CG_{in}$] and tpv_j = [Ref, $CG_{i1}, \ldots, CG_{im}$] are *tuple types*; then $TPt_i =_t TPt_j$ iff n = m and $\forall x\ CG_{ix} =_t CG_{jx}(1 \le x \le n)$; $TPt_i =_T TPt_j$ iff n = m and $\forall x\ CG_{ix} =_T CG_{jx}(1 \le x \le n)$;

Definition 3.8. The equation "$=_v$" and "$=_V$" between values in dynamic table are defined as follow:

attribute value: If $av_i = <A_i, v_i>$ $av_j = <A_j, v_j>$ then $av_i =_v av_j$ iff $v_i =_t v_j$; $av_i =_V av_j$ iff $A_i = A_j \wedge v_i =_v v_j$

attributes group value: If $agv_i = (av_{i1}, \ldots, av_{in})$ $agv_j = (av_{j1}, \ldots, av_{jm})$ are *attribute values*; then $agv_i =_v agv_j$ iff n = m and $\forall x\ av_{ix} =_v av_{jx}(1 \le x \le n)$; $agv_i =_V agv_j$ iff n = m and $\wedge \forall x\ av_{ix} =_V av_{jx}(1 \le x \le n)$

tuple value: If $tpv_i = [o_i, N_{i1}:agv_{i1}, N_{i2}:agv_{i2}, \ldots, N_n:agv_{in}]$ and $tpv_j = [o_j, N_{j1}:agv_{j1}, N_{j2}:agv_{j2}, \ldots, N_{jm}:agv_{jn}]$ are *tuple values* then $tpv_i =_V tpv_j$ iff n = m and $\wedge N_{ix} = N_{jx} \wedge \forall x\ agv_{ix} =_V agv_{jx}(1 \le x \le n)$

The equation "$=_t$" and "$=_v$" only need the corresponded types and values of the attributes are equivalent while "$=_T$" and $=_V$ need the name of the corresponded attributes must be the same too. For example, (name:string, size:integer) $=_t$ (N:string, S:integer) but (name:string, size:integer) $\neq_T$ (N:string, S:integer).

Selection Operator

Since the schema of the tuple can be different in a dynamic table, the selection operator is more complicated compared to the one in 1NF relation. The select predicate includes not only the comparisons between the values but also the conditions concerned to the schemas:

Definition 3.9. Select Function(SF) is defined as follow:

If $N_i.A_{ix}$ and $Nj_\circ$ Ajy (N_i and N_j are two column group names)are two attributes of the same type in a DTs, *c* is a constant *attribute value* of $N_i.A_{ix}$, then $N_i.A_{ix}$ cop $N_j.A_{jy}$ and $N_i.A_{ix}$ cop *c* (cop$\in\{<, \le, \ge, >, =\}$(if *c* is an *identifier value* cop = "=")) are SFs;

If $CG_i = <N_i, Cs_i>$and $CG_j = <N_j, Cs_j>$ are two *column types*, *c* is a *column group type* or an *attributes group value*, then N_i eq N_j(eq$\in\{=_t, =_T, =_v, =_V\}$) and N_i eq *agv*(eq$\in\{=_t, =_T, =_v, =_V\}$) *agv* are SFs;

If $CG_i = <N_i, Cs_i>$ is a *column group type* and $C_{ix} = <A_{ix}, B_{ix}>$ is a *column type*, then exist(CG_i) and exist($N_i.A_{ix}$) is a SF;

If S is a SF, then (not S) is a SF;

Definition 3.10. If DT is a table, a select operation is denoted by $\sigma[F](DT)$, F is a SF; According to the type of SF

SF is type (a), then $\sigma[F](DT) = \{tp | tp \in DTv \wedge (tp[N_i.A_{ix}]$ cop $tp[N_j.A_{jy}])\}$; $\sigma[F](DT) = \{tp | tp \in DTv \wedge (tp[N_i.A_{ix}]$ cop $c)\}$ (if *c* is an *identifier value* cop = "="));

SF is type (b), then $\sigma[F](DT) = \{tp|tp \in DTv \wedge (tp[N_i]\ eq\ tp[N_j])\}(eq \in \{=_t, =_T, =_v, =_V\})$; $\sigma[F](DT) = \{tp|\ tp \in DTv \wedge (tp[N_i.A_{ix}]\ eq\ c)\}(\ eq \in \{=_t, =_T, =_v, =_V\})$;

SF is type (c), then $\sigma[F](DT) = \{tp \in DTv|\ (N_i \in AttG(tp) \wedge tp[N_i.A_{ix}] = True|Null)\}$; $\sigma[F](DT) = \{tp \in DTv|\ (N_i.A_{ix} \in Attr(tp) \wedge tp[N_i.A_{ix}] = True|Null)\}$;

SF is of type (not F′) then $\sigma[F](DT) = \{tp \in DTv\ |\sigma[F'](DT)=False\}$;

According to Definition, the comparison between types can be a predicate in selection operator of dynamic table, so the first tuple will be selected [id1, BasicInfo:[name: Flower, size: 128k], Semantic:[Content: Flower Tone: Yellow]] with the selection σ [BasicInfo.name = "Flower"](DT) and σ [BasicInfo $=_t$ [CG:(A1:string, A2:integer)]](DT).

4 Implementation of Dynamic Table and Intro to Muldas

In this section, we introduce the how this upper layer structure are implemented on our key-value based Muldas system.

Massive Unstructured cLoud Data mAangement System – Muldas is accomplished by the Key Laboratory of Machine Perception (Peking University) based on Hadoop and Map-reduce framework using the data model of the dynamic table. Like other popular cloud data storage systems, Muldas is also based on the key value pattern; that is to say each cell of a tuple is a key-value pair. The main difference between Muldas and those systems is how the key-value pairs are organized physically.

In dynamic table, columns are separated into groups; user of Muldas can choose the physical format for each column group respectively: row-oriented or column-oriented; the key-value pairs of a same row or column are authentically stored together physically and this is the biggest difference between systems like Hbase or Cassandra.

In Muldas, the cells in row-oriented column groups are stored in the following format.

Row oriented:

Column Group	Row ID	Column	Value

Column oriented:

Column Group	Column	Row ID	Value

The keys of both kinds of cells consist of Row ID, Column and Column Group but have different orders. For the cells in row oriented, the Row ID is placed before the column, which is reversed in column-oriented format.

The key-value pairs are stored in the underlying HFile in HDFS and compressed with LZO algorithm according to keys in dictionary order. Since the column group tag has the highest priority, cells in different column group are separated. With the row ID prior to the column, cells from one tuple are organized in row-oriented column groups, same as the ones in column-oriented column groups.

We use a meta table to store the schema, type information and physical format of dynamic tables; to save more spaces, we store an integer id instead of storing the raw string of column and column group name, so the meta table also holds name-to-id relationships. With this strategy, any cell in the dynamic table can be easily decoded to get the column group, row and column it belongs to and physical format and data type value it holds.

Figure 5 shows how a dynamic table on logical layer are stored in the physically in key-value pairs.

Row ID	Column Group 1 (column-oriented)			Column Group 2 (row-oriented)			
	c1	c2	c3	c4	c5	c6	c7
1001	v_{101}	v_{102}	Null	Vacant	Vacant	Vacant	Vacant
1002	v_{201}	v_{202}	v_{203}	Vacant	Vacant	Vacant	Vacant
1003	Vacant	Null	v_{303}	v_{304}	Null	v_{306}	v_{307}
1004	Vacant	Vacant	Vacant	v_{404}	v_{405}	v_{406}	Null

Dynamic table on logical layer

Key			Value
01	11	1001	v_{101}
01	11	1002	v_{201}
01	12	1001	v_{102}
01	12	1002	v_{202}
01	12	1003	Null
01	13	1001	Null
01	13	1002	v_{203}
01	13	1003	v_{303}
02	1003	21	v_{304}
02	1003	22	Null
02	1003	23	v_{306}
02	1003	24	v_{307}
02	1004	21	v_{404}
02	1004	22	v_{405}
02	1004	23	v_{407}
02	1004	24	Null

Layout of the key-value pairs stored in HFile

Name	ID	Type
Column Group 1	01	Column-Oriented
Column Group 2	02	Row-Oriented
C_1	11	Int
C_2	12	String
C_3	13	Double
C_4	21	Float
C_5	22	Char
C_6	23	BigInt
C_7	24	Date

Schema information in meta table

Fig. 3. Physical Data Layout of A Logical Dynamic Table in HDFS

Based on the algebra defined above, Muldas provides a SQL-like structured query language named MQL which is not supported in the typical No-SQL systems like Hbase and Cassandra. MQL supports most SQL-like DDL and DML statements, including createtable, droptable, select, insert, project, deleterow, update and even join operations. In MQL, a column is located by the path:

TableName.ColumnGroupNname:ColumnName;

Users can write MQL queries just like writing a normal SQL queries. Besides, MQL support schema modification operations: addcolumn, deletecolumn, addcolgroup, delcolgroup defined in section 3.

5 Experiment Evaluation

In this section, we will evaluate our proposed Dynamic Table for many concerned operations on both wide sparse data and dense data. By making a comparison with Muldas to the state-of-art systems, we examine the comparative advantages of row-stores and column- stores for Dynamic Table respectively.

5.1 Experimental Environment and Dataset

Concretely, a real-word dataset and a synthetic dataset are used to evaluate the experiments results. The data sets are about 100G in text format from the internet available at (http://an.kaist.ac.kr/traces/WWW2010.html) containing about 300 attributes and 832,606,357 tuples. The data sets can be divided into three parts: (I) the first data set is student course selection information while the courses are divided into 32 courses required and 226 elective courses, a student is asked to take almost all the required courses and a certain number of elective courses. Every student will choose 28 elective courses on average; (II) the second part are student basic information including: student id, name, age, gender, birthday, home town. There are about 40% absent values in each column except the student id column; (III) the third part covers the hobby of the student including: favorite music style, movie, games and sports which is very sparse that almost 70% are absent values. In this experiment, the data sets are divided into four column groups: CG1(Required Courses), CG2(Basic Info), CG3(Hobbies) and CG4(Elective Course), listed in the descending order by the absent value occupancy.

As for the baseline systems, we chose cloud data stores as follows, (1) HBase-0.90.2[10], an open source No-SQL distributed database modeled after Google's BigTable[12]. It is on top of Hadoop and supports random reads and random writes; (2) Hive-0.6.0, an open source data warehousing solution built on top of Hadoop, the files of which can be selected to be stored as RCFile[7]; (3) Pig Latin with all the data stored in LZO compressed text file in row-oriented or column oriented format; (4) Muldas.

In consideration of the limited resources, we deploy all 4 systems on the same cluster with 20 nodes connected by a 1Gbit Ethernet switch. Each node has 4 cores (Quad-Core AMD Opteron(tm) Processor 2378MHz), 16GB of main memory, and 1.4TB SATA disks.

5.2 Results and Discussions

To compare the respective advantages and display the differentiated storage requirements, we would like to validate the performance of Dynamic Table from the following aspects:

Data Loading Time and Storage Space Utilization
The first experiment is to evaluate the time efficiency and the space utilization. We import the data sets into related systems. In the data loading time experiment, we take sample with the same sampling rate for each part of the data to evaluate the performances of each system with different data size. And for Muldas, we load the data in

different storage formats to show how it will affect the performance. The result is in Figure. Muldas shows a better performance when it uses the column storage format for the sparse part of the data.

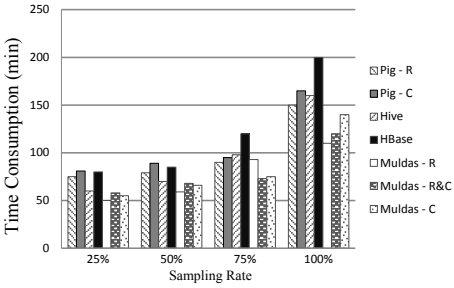


Fig. 4. Data Loading Consumption

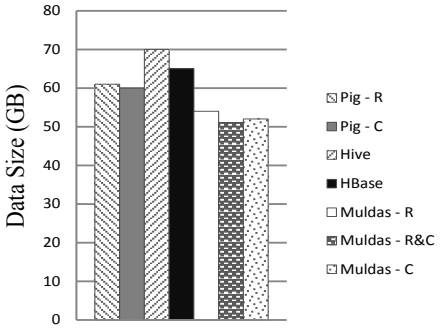


Fig. 5. Space Utilization

The storage space utilization result shows that the databases that using a traditional related model and row-oriented physical format cost a lot of space.

Query Performance

In this experiment, we use one very typical query that is very common in date retrieve. The example query is (written in MQL):

```
select * from T1 where CG1.math > 90 AND CG2.age < 30 OR
CG3.major ==”computer” AND CG4.CloudComputing < 60
```

Since Pig and Hbase do not have such kind or structured query language, we transform the query into the equivalent Pig Latin and Hbase API to show the comparison result. We also give the comparison of the length of these equivalent query codes.

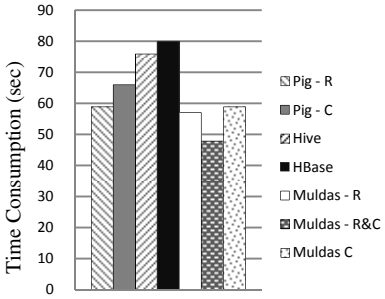


Fig. 6. Query Performance

Pig Latin	UDF for row-oriented	UDF for column-oriented
	34 lines	103 lines
Hive HQL	2 lines	
Hbase API	85 lines	
Muldas MQL	2 lines	

Fig. 7. Query Code Length

As shown in the result, Muldas shows a good query performance because the storage strategy and the data density are taken into account. Hive is very slow when there are many sparse data in the dataset. Although Hbase is stored logically in column, the

data in one column family are actually stored according the order the insert time of each cell so data in one column maybe separated. The result also shows that with Hbase and Pig, it's very inconvenient to do the query because you have to write the User Defined Function for Pig (especially complex for the data stored in column) and the API for Hbase to get the data in each cell.

6 Conclusions

In this paper, we try to address the problem of setting up a flexible storage structure for datasets with different characteristics in the cloud data store. The goal is to offer a hybrid row and column layout to satisfy the differentiated requirements of wide sparse data and dense data. A new NF^2 scalable storage structure named "Dynamic Table" based on the key-value storage is proposed. The formal definition of dynamic table and the algebra are given. We also implement the dynamic table in an unstructured data management system – Muldas based on Hadoop. The experiment on real data shows the comparison result between our Muldas system and data storage on cloud.

References

1. Gantz, J.F.: The Expanding Digital Universe. International Data Corporation (2007)
2. Doan, A., Naughton, J.F., Baid, A., et al.: Information extraction challenges in managing unstructured data. ACM SIGMOD Record Archive 37(4), 14–20 (2008)
3. Lux, M., Chatzichristofis, S.A.: LIRe: Lucene Image Retrieval – An Extensible Java CBIR Library. In: MM 2008 Proceedings of the 16th ACM international conference on Multimedia, pp. 1085–1088 (2008)
4. Tamura, H., Mori, S., Yamawaki, T.: Textural features corresponding to visual perception. IEEE Transactions on Systems, Man, and Cybernetics 8(6), 460–472 (1978)
5. Hirata, K., Kato, T.: Query by Visual Example - Content-Based Image Retrieval. In: Pirotte, A., Delobel, C., Gottlob, G. (eds.) EDBT 1992. LNCS, vol. 580, pp. 56–71. Springer, Heidelberg (1992)
6. Apache Hive, `http://hive.apache.org/`
7. He, Y., Lee, R.B., Huai, Y., et al.: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems. In: Proceedings of the IEEE International Conference on Data Engineering (ICDE), pp. 1199–1208 (2011)
8. Beckmann, J.L., Halverson, A., Krishnamurthy, R., et al.: Extending RDBMSs to support sparse datasets using an interpreted attribute storage format. In: Proceedings of the 22nd International Conference on Data Engineering ICDE, pp. 58–74 (2006)
9. Abadi, D.J.: Column Stores For Wide and Sparse Data. In: Proceedings of CIDR, pp. 292–297 (2007)
10. Apache HBase, `http://hbase.apache.org/`
11. Apache Cassandra, `http://cassandra.apache.org/`
12. Chang, F., Dean, J., Ghemawat, J., et al.: Bigtable: A Distributed Storage System for Structured Data. J. ACM Transactions on Computer Systems 26, 1–26 (2008)
13. Apache Pig, `http://pig.apache.org/`
14. `http://www.mongodb.org/`
15. `http://redis.io`

Reflection on the Popularity of MapReduce and Observation of Its Position in a Unified Big Data Platform

Xiongpai Qin[1,2], Biao Qin[1,2], Xiaoyong Du[1,2], and Shan Wang[1,2]

[1] MOE Key Lab of Data Engineering and Knowledge Engineering, Beijing, 100872, China
[2] School of Information, Renmin University of China, Beijing, 100872, China
{qxp1990,qinbiao,duyong,swang}@ruc.edu.cn,
qxp1990@sina.com

Abstract. In recent years MapReduce has risen to be the de-facto tool for big data processing. MapReduce is a disruptive innovation. It has changed the landscape of database market, the landscape of technologies, as well as the landscape of saying power. The article will give a reflection on the popularity of the technique and some observations of its position in a unified big data platform.

Keywords: MapReduce, Popularity, Reflection, Unified Big Data Platform.

1 The Popularity of MapReduce

MapReduce, introduced by Google in 2004[1] as a tool for huge volume of unstructured data processing, is rising to become the de-facto tool for big data processing. Industry has first recognized the value of MapReduce, and dozens of Hadoop (an open source implementation of MapReduce) based startups are launched to provide big data processing, analysis, and visualization solutions. To name a few of them, they are Cloudera, HortonWorks, MapR, Karmasphere, DataMeer, Aster Data, Greenplum, Hadapt, and Platfora. The startups share a common characteristic, they built their businesses upon the MapReduce technology. The strong wind of MapReduce swept through parallel computing community and aroused a tide of research during the period of 2006 to 2008, as well as another tide of research in database community during the period of 2009 to 2012.

The research has improved MapReduce in many aspects [2] [3], including: (1) Storage layout & data placement optimization, handling of data skew, index support, and data variety support; (2) Extension of MapReduce for stream processing, incremental processing, iterative processing; (3) two way join, multi way join, theta join optimization, parallelization of data mining/machine learning algorithms; (4) Schedule strategies for multi core CPU, GPGPU, heterogeneous environment, and cloud; (5) Easy to use interfaces for SQL query, statistical, data mining & machine learning algorithms, such as Hive, Pig, System ML, and Mahout; (6) Energy saving techniques, private and security guarantee for MapReduce.

The computing model of MapReduce is rather simple. MapReduce is a general execution engine that is ignorant of schemas and storage models, i.e. there is no

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 339–347, 2013.

structure to the data. The runtime system automatically parallelizes computing tasks across a large cluster of commodity hardware, handles failures and manages disk I/O & network efficiency. The user only needs to provide a map function (which is applied to all input rows of the dataset to produce an intermediate output) and a reduce function (who will aggregate the intermediate results to produce the final result). However, MapReduce is not that simple. MapReduce can do analytic works beyond computing statistics from up to PB of data, or simply performing ETL (extract, transform, and load) work. Various data analytics algorithms have been migrated onto the MapReduce platform, including data summarization and reporting, multi-dimensional analysis and OLAP, data mining, machine learning, information retrieval, text mining and sentiment analysis, science data processing, as well as social network analysis and general graph analysis. The performance of the algorithms is improved, and application of the MapReduce technology has expanded to many domains. Some people think that such works are reinventing the data warehousing and BI techniques that have been created over the years, the authors believe that the so-called "reinventions" are necessary and may lead to important innovations.

With the growing influence of MapReduce, Traditional vendors in the data market noticed the popularity of MapReduce. IBM moves quickly with its *Big Insights* Plan, the plan tries to integrate DB2, Netezza, Hadoop, and SPSS into a big data analytic platform. EMC, formerly not as a database vendor, became a big player in the market overnight through acquiring Greenplum, which has combined the PostgreSQL database, shared nothing architecture and MapReduce to create the core product. TeraData acquired Aster Data to obtain its experience of MapReduce engineering as well as the analytic software package using MapReduce-style parallelism. Most funny is that, several venders who looked down on MapReduce before finally change the tough minds, Microsoft rejected MapReduce in 2009, in the year of 2012 it has closed the Dryad project (an equivalent parallel computing framework of MapReduce by Microsoft) [4] and warmly embrace Hadoop; Oracle despised MapReduce early in 2011, eventually published its *Big Plan* late in 2011, which incorporates noSQL and Hadoop technologies into the whole picture.

The penetrating power of MapReduce has changed the landscape of market, i.e. traditional venders vs. dozens of startups, the landscape of technology, i.e. RDBMS vs. noSQL/Hadoop, as well as the landscape of saying power, we see that many researchers previously not in the database community now rush into the field of data processing and analytics, and bring forth their innovative ideas, which deserve serious notice.

2 Reflection – Why MapReduce Is So Popular

With the price of the storage device going down and the capacity of it increasing, development of internet applications (including e-commerce, social network …), the requirements of science simulation and research etc., it is the first time that people collect some data sets with huge volumes. The era of Big Data is coming. To describe big data, people have agreed on three *V*s – *Volume, Variety, and Velocity*. Firstly, the volume of the data has exceeded the capability of traditional data processing tools (specifically RDBMS, relational database management systems). Secondly, people

need to handle various types of data to extract insightful information for decision making. The data is essentially multi-structured, including semi-structured data, unstructured data as well as structured data. Finally, the speed of data generation is very fast, the data emitted from sensors can be taken as an example, which needs to be handled in a timely manner.

The three Vs raise a range of challenges, among which the volume is the dominant one, followed by data variety. On one hand, handling various types of data of big volume with high velocity is beyond traditional tool's capability, on the other hand, although RDBMS technology has advanced dramatically over the past several decades, the computing paradigm are shifting from big server-based computing to distributed computing on clusters of low-cost commodity hardware for cost reason. MapReduce was born at the right time.

2.1 The Debate and the Outcome

Some prominent database researchers argue that there is nothing new about systems like MapReduce and that they're actually a step backward [5]. The argument has aroused a fierce debate. Is that true? The answer is no. The Apache Hadoop open source software project won the top prize of 2011 Media Guardian Innovation Awards [6], described by the judging panel as a "*Swiss army knife of the 21st century*".

Actually database researchers should be more open-minded. Sometimes the strong sense of glory and pride that has been brought about by the huge success of RDBMS will blind their eyes. The rising of instead of dying out of MapReduce has been beyond their expectation, and most annoying is that MapReduce begins to expand its application from web search to territories used to be occupied by relational database systems, including OLAP, data mining, machine learning, and information retrieval. Hadoop has become the winner of the ***Jim Gray prize*** for the fastest sort of a terabyte of data. In 2008, Hadoop required 209 seconds to analyze the terabyte, using a 900 node cluster. In 2009, it won the prize for an analysis that took only 62 seconds, running on 1,500 nodes. Some criticisms on MapReduce are not so rational, conversely the criticisms justify the promise of MapReduce technology.

The authors would like to clarify some wrong saying about MapReduce, including:

(1) MapReduce is Map function plus Reduce function: No, that is not true. MapReduce is a general parallel computing framework for big data processing. It includes the highly fault tolerant distributed file system, the MapReduce parallel computing model, and the running time that taking care of parallel running of applications on a large cluster. The whole MapReduce ecosystem has several more peripheral add-ons.

(2) MapReduce can only do some batch processing: No, that is not true. After Google published its Dremel [7]. Several venders are trying to provide interactive ad-hoc query processing capability onto Hadoop, including Cloudera *Impala*, HortonWorks *Stinger*, Apache *Drill*, and EMC *HAWQ*. Dremel achieves interactive ad-hoc query capability on big data by combining multi-level execution trees and columnar data layouts. It is capable of running aggregation queries over trillion-row tables in seconds. Dremel can scale to thousands of CPUs and PB of data. Now the technology has been used by thousands of users inside Google.

(3) MapReduce can only handle un-structured data, or is more suitable for un-structured data handling. No, that is not true. One example to counter argue the point is the RCFile [8] of FaceBook (together with Ohio State University and Chinese Academy of Science). RCFile is a storage layout optimization for Hadoop data blocks for high performance of structured data processing. The work of RCFile is inspired by the PAX idea from RDBMS research, it applies an elaborate structure to HDFS (Hadoop Distributed File System) blocks to achieve higher performance of data accessing while fully retaining nice properties of scalability and fault tolerance of MapReduce. In RCFile, firstly data (structured table) is horizontal partitioned into blocks. In each block, the data is further broken into columns and every column is store contiguously in the block. Compression techniques can be used to reduce space consumption because the data is stored in blocks using a columnar layout. RCFile has been deployed to production by FaceBook, and plays an important role in daily Facebook operations. From the example, we can see that, by applying some elaborate structure to HDFS blocks, MapReduce can handle structured data as well. Actually the MapReduce framework is ignorant of the underlying storage layout of data blocks, it depends on applications to interpret the data at run time, no matter what the storage layout of the data is.

2.2 Reflection – Application (User) Requirement is the King

Taking TV as an example, we would like to comment that - the user requirement is finally the king, not the technologies. People have the desire to entertain themselves with Audio and Video devices. TV is a good product for that purpose. TV technology has evolved from CRT, to LCD, and then to LED recently, the technology keep changing, but the desire of people for better playback of Audio and Video never change, and technologies just serve the purpose of entertainment and information acquirement of people. Technologies are means, not objectives.

Similarly, when the big data era is coming and RDBMS can not completely address the challenges, MapReduce can be an alternative/complementary tool to RDBMS. What people need is to handle various types of data (structured data, unstructured data, and semi-structured data), data at different stages (data in motion, data at rest, and archived data) to extract useful insights, what tools are used to do the job doesn't matter in some sense. Actually the preferred architecture model for web scale data processing of parallel computing over large scale low cost server-based clusters has set off an innovation path that outpace the traditional database market.

Recently what is a really pressing issue is to blend the two technologies of RDBMS and Hadoop for multi-structured data analytics (next section).

3 Positioning MapReduce in a Unified Big Data Platform

RDBMS has been studied for more than 40 years, optimization techniques across different levels from storage layer to execution layer have been investigated, and many of them have been implemented in RDBMS. An ecosystem (vendors, products,

tools, services…) built around RDBMS has been around for several decades. MapReduce was born for large dataset processing, it is designed with highly scalability and highly fault tolerance as top considerations. A big data analytics ecosystem built around MapReduce is emerging alongside the traditional one built around RDBMS.

When it comes to data analytics, the objectives of RDBMS and MapReduce as well as the ecosystems built around them, overlap much really, in some sense they do the same thing and MapReduce can even accomplish more works, such as graph processing, which RDBMS can not handle well. Why two techniques/ecosystems for one thing? People are envisioning convergence of the two.

While traditional venders such as IBM, Microsoft, TeraData and Oracle reestablish their positions in the big data market by adopting Hadoop, so many startups such as EMC (Greenplum), HortonWorks, Cloudera etc. will cut off a large portion of the market. The big data platforms they provide fall in three categories: (1) *Co-Exist* solutions; (2) *SQL with MapReduce Support* solutions; and (3) *MapReduce with SQL Support* solutions.

3.1 Co-exist Solution – Different Tools for Different Jobs Plus High Speed Data Integration

The big data solutions provided by IBM and Oracle fall into this category.

IBM is a fast mover in the big data era. Its Watson computer, which has won the game of Jeopardy in 2011, was powered by Hadoop technology. IBM has announced its *Big Insights* in 2010. Since IBM has so many data management and processing product lines, it is natural to combine these underlying technologies to provide a big data processing environment, which is called *Big Data Platform* by IBM [9].

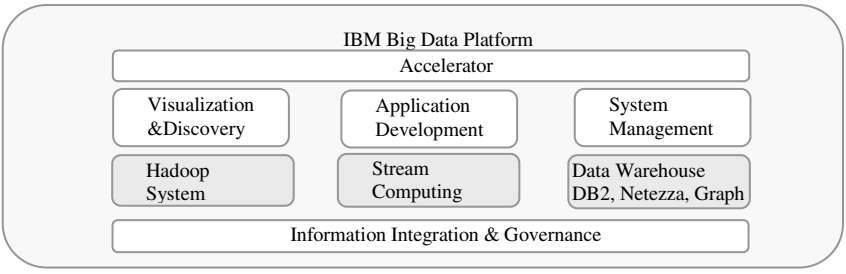


Fig. 1. The Architecture of IBM's Big Data Platform [9]

There are several components in the IBM Big Data Platform, including basic data management and processing engines such as Hadoop system, Stream system, and Data warehouse system, visualization & discovery tools, applications development tools, system management tools, as well as information integration and governance facilities. For IBM, the big data analysis is viewed as a workflow of analytic tasks instead of a single analytic task. These analytical processes can run across multiple tools in the Big Data Platform to process multi-structured data (both structured and un-structured data).

Oracle was rather reluctant to accept noSQL & Hadoop in the early of 2011. In the end of 2011, under the great pressure of EMC and TeraData, as well as many other data warehouse startups that based their business on the Hadoop technology, Oracle released its *Big Plan.* In the *Big Plan*, Oracle adopts a combination and co-exist strategy [10], which resembles IBM's Big Data Platform.

IBM and Oracle provide connectors to facilitate the process of data integration, including tools for loading data from Hadoop into database, accessing HDFS data from database, transparent access to Hadoop from statistical package etc.

3.2 SQL with MapReduce Support

Microsoft PolyBase, EMC Greenplum, TeraData Aster Data fall in this category.

In PolyBase [11], Users can create external tables on HDFS data. This allows queries to reference data stored in HDFS as if it were loaded into a relational table. Users can seamlessly perform joins between tables in PDW and data in HDFS. PolyBase use a cost based optimizer to determine on whether the job is done on PDW or on Hadoop HDFS according to some statistics gathered before. Future version of PolyBase will further utilize the huge computation and I/O power of the Hadoop cluster by offloading more work to Hadoop for processing, even for queries that only reference PDW-resident data.

EMC became a big player in the market by acquiring Greenplum. In the whole picture of EMC *Unified Analytics Platforms (UAP),* Greenplum DATABASE and Greenplum HD, two data storage and processing engines, are working in a unified and interaction manner to co-process all of user data. The Greenplum DATABASE is built upon PostgreSQL database for structured data processing. Greenplum HD is responsible for unstructured data processing with Hadoop. Actually with the *HAWQ* technique, Greenplum HD can perform SQL queries over Hadoop resident data. *Dynamic Pipelining*™ technology, together with many other implementation optimizations, deliver a 100X to 600X performance improvement over other interactive Hadoop SQL solutions such as *Impala* of Cloudera according to EMC [12].

To response to the EMC acquirement of Greenplum, TeraData acquired Aster Data to maintain its competitive edge in the market. TeraData believes that different data carries different value, comes in different formats, requires different analytics, and serves different business objectives [13]. It is reasonable to utilize multiple technologies, including: data staging, data discovery, data warehouse to processing the data. TeraData integrates open source Hadoop, Aster Data discovery platform, and TeraData data warehouse into the *TeraData Unified Data Architecture* ™, which act as the staging tool, the discovery tool, and the data warehouse respectively. The whole architecture resembles IBM's Big Data Platform.

But wait a minute, when we take a closer look at it, we can see that Aster Data database is similar to EMC *Unified Analytics Platforms (UAP)* in itself. Aster Data database is a massively parallel (MPP) analytic platform, both SQL and MapReduce analytic processing capabilities are embedded in the product through *SQL-MapReduce*® [14] technology. Aster Data database can seamlessly access multi-structured data in database table and HDFS. Actually Aster Data database can be a replacement of TeraData database, used as a standalone data processing platform.

3.3 MapReduce with SQL Support

The Apache *Drill* project, solutions of startups such as HortonWorks, Cloudera, Hadapt, Platfora, as well as next generation of Hadoop fall in this category.

In a small territory, HortonWorks and Cloudera are competitors, because they all provide enhanced Hadoop solutions for enterprise adoption. In the basic Hadoop ecosystem, there are common components such as HDFS, MapReduce, HBase, Pig, Hive, Meta Data Services, Management and Monitoring Services, as well as Data Integration Services. Besides basic components, HortonWorks and Cloudera provide add-on tools to facilitate enterprise adoption of Hadoop as the de-facto big data platform, including high available capability, management and operation tools.

As for SQL support, HortonWorks and Cloudera introduce *Stinger* and *Impala* respectively. *Stinger* and *Impala* are similar in that they try to provide the interactive SQL-like query capability over Hadoop, as a replacement or enhancement of Hive. Apache Hive and its HiveQL language have become the standard SQL interface for Hadoop since introduced by Facebook in 2007. Hive is often criticized for its low performance. But now the situation is changing.

Stinger is an initiative launched by HortonWorks, which is to make Hive much faster, so that people can run ad-hoc query on Hadoop interactively. HortonWorks plans to realize the *Stinger* in a few steps. Firstly, they are making Hive a more suitable tool for people to perform decision support queries on Hadoop by adding new features to the language and making Hive's system a model more like the standard SQL. Secondly, they are improving the Hive's query optimizer for better query execution plans, and Hive's execution engine for higher throughput of data processing. Thirdly, they develop a new columnar file format for higher performance of analytic tasks. Lastly, they are introducing a new runtime framework - *Tez*, to reduce Hive's latency and throughput constraints [15].

Cloudera tries to push Hadoop beyond a batch analytic tool to an interactive analytic tool with *Impala*. *Impala* is a real-time query engine that allows users to query data stored in HDFS and HBase in seconds via a SQL interface. Instead of using MapReduce, *Impala* uses its own processing framework to execute queries over HDFS, which results in a 10x-50x performance boost compared to Hive.

Apache *Drill* is design to scale to 10,000 servers or more and to be able to process Petabytes of data and trillions of records in seconds. The *Drill* architecture consists of four key layers [16]: (1) Query languages: This layer is responsible for parsing the user query and constructing an effective execution plan. *Drill* is designed to support SQL-like language (called DrQL) as well as other languages; (2) Low-latency distributed execution engine: The layer provides the scalability and fault tolerance needed to query Petabytes of data efficiently on a large cluster; (3) Data formats: various data formats, including column-based format, schema-less formats, and row-based formats are to be supported in this layer; (4) Data sources: Hadoop HDFS etc.

The interactive capabilities that these tools provide will refute the bias saying that "Hadoop can only do some batch processing of data".

Hadapt is the commercialized version of the HadoopDB [17] of Yale University. Hadapt combines the scalable architecture of Hadoop with a hybrid storage layer that integrates a relational data store and HDFS, by consolidating data into a single

platform Hadapt eliminates the need to move around data inside the system. The *Hadapt Interactive Query* tries to deliver interactive capability on Hadoop, like Cloudera *Impala*, Apache *Drill*, EMC Greenplum *HAWQ*, and HortonWorks *Stinger*. Hadapt also provides *Hadapt Development Kit*™ (HDK) for easier application development.

Platfora is a startup launched in the late of 2012, which deserves serious notice. After series A and series B fund securing, it has drawn in a total of more than $25.7M. The great ambition of Platfora is to make Hadoop an interactive big data platform by using in memory data processing (aggregation and cache) technology, data compression, and columnar data layouts. Platfora is a big data solution completely built upon Hadoop [18].

The current Apache Hadoop MapReduce framework has several limitations, such as the scalability limit of around 4,000 machines. Next generation of Hadoop is on the horizon. The major change to current version of Hadoop is to separate the two functions of the Job Tracker (resource management and job scheduling/monitoring) into different components. The new Resource Manager manages the global assignment of compute resources to applications, and the Application Master manages the application's scheduling and coordination [19]. The re-architecture leads to a new framework that scales out more easily. After decoupling MapReduce computing paradigm from the resource management, new application types can be plugged into Hadoop, including stream processing, graph processing, bulk synchronous processing (BSP, for graph data processing) and message passing interface (MPI, many parallel computing applications use the MPI computing model) [20]. Next generation of Hadoop will be greatly enhanced in terms of scalability, availability and performance. And it is expected to be the underpinning technology for Hadoop-based unified big data platforms, competing with RDBMS-based ones. The days RDBMS dominates the whole market has gone.

4 Conclusions

When people need to analyze big volume of multi-structured data for insightful information, Hadoop is an indispensable component in a unified big data platform. There are *Co-Exist, SQL with MapReduce Support, MapReduce with SQL Support* solutions for combining RDBMS and MapReduce. RDBMS and Hadoop are the tools to serve our goal of data management and analytics, user requirements are the ultimate forces that drive the innovation, tools in themselves cannot be the objectives. Database community should be open to incorporate innovative ideas (such as MapReduce…) from other sub-disciplines of computer science (such as parallel computing…) to tackle the challenges of big data processing.

Acknowledgements. This work is partially funded by the NSF of China under Grants No. 61170012 & 61170013, the Funds of Renmin University of China under Grant No. 10XNJ048, and EMC Global CTO Office.

References

1. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. In: Symposium on Operating Systems Design and Implementation (OSDI), pp. 137–150. USENIX Association, San Francisco (2004)
2. Lee, K.H., Lee, Y.J., Choi, H., Chung, Y.D., Moon, B.: Parallel data processing with MapReduce: a survey. SIGMOD Record 40(4), 11–20 (2011)
3. Sakr, S., Liu, A., Fayoumi, A.G.: The Family of MapReduce and Large Scale Data Processing Systems (2013), http://arxiv.org/abs/1302.2966
4. Foley, M.J.: Microsoft drops Dryad; puts its big-data bets on Hadoop (2011), http://www.zdnet.com/blog/microsoft/microsoft-drops-dryad-puts-its-big-data-bets-on-hadoop/11226
5. Kraska, T.: Finding the Needle in the Big Data Systems Haystack. IEEE Internet Computing 17(1), 84–86 (2013)
6. Winckler, M.: Apache Hadoop takes top prize at Media Guardian Innovation Awards (2011), http://www.guardian.co.uk/technology/2011/mar/25/media-guardian-innovation-awards-apache-hadoop
7. Melnik, S., Gubarev, A., Long, J.J., Romer, G., Shivakumar, S., Tolton, M., Vassilakis, T.: Dremel: Interactive Analysis of WebScale Datasets. Proceedings of the VLDB Endowment 3(1-2), 330–339 (2010)
8. He, Y., Lee, R., Huai, Y., Shao, Z., Jain, N., Zhang, X., Xu, Z.: RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems. In: International Conference on Data Engineering (ICDE), pp. 1199–1208. IEEE Computer Society, Hannover (2011)
9. Ferguson, M.: Architecting a Big Data Platform for Analytics. A Whitepaper Prepared for IBM (2012)
10. Oracle: Oracle: Big Data for the Enterprise. Oracle White Paper (2012)
11. Dewitt, D.: Polybase: What, Why, How. SQL PASS Summit Keynote (2012)
12. EMC: Unified Analytics Platform (2013), http://www.greenplum.com/products/greenplum-uap
13. TeraData: TeraData Unified Data Architecture. TeraData Whitepaper (2012)
14. Friedman, E., Pawlowski, P., Cieslewicz, J.: SQL/MapReduce: A practical approach to self describing, polymorphic, and parallelizable user defined functions. Proceedings of the VLDB Endowment 2(2), 1402–1413 (2009)
15. Gates, A.: The Stinger Initiative: Making Apache Hive 100 Times Faster (2013), http://hortonworks.com/blog/100x-faster-hive/
16. Incubator Wiki: Drill Proposal (2013), http://wiki.apache.org/incubator/DrillProposal
17. Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Silberschatz, A., Rasin, A.: HadoopDB: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads. Proceedings of the VLDB Endowment 2(1), 922–933 (2009)
18. Platfora: Platfora Homepage (2013), http://www.platfora.com/
19. Murthy, A.C.: The Next Generation of Apache Hadoop MapReduce (2011), http://developer.yahoo.com/blogs/hadoop/posts/2011/02/mapreduce-nextgen/
20. BUSINESS WIRE: HortonWorks to Deliver Next-Generation of Apache Hadoop (2012), http://www.businesswire.com/news/home/20120119005825/en/Hortonworks-Deliver-Next-Generation-Apache-Hadoop

Cleaning Missing Data Based on the Bayesian Network

Liang Duan, Kun Yue*, Wenhua Qian, and Weiyi Liu

Department of Computer Science and Engineering,
School of Information Science and Engineering, Yunnan University, 650091, Kunming, China
kyue@ynu.edu.cn

Abstract. To guarantee the data quality, it is necessary to clean the missing data that prevalently exist in real world databases. By incorporating additional information, such as functional dependencies or integrity constraints, the correct value for each missing data item can be derived in many existing data cleaning methods. In this paper, we propose a method for cleaning the missing data item without additional information by adopting Bayesian network (BN) as the framework of the representation and inferences of probability distributions. First, we learn a Bayesian network from the complete part of the given incomplete database, called IBN. Then, we infer the probability distributions of each missing data item based on Gibbs sampling upon the IBN. Consequently, we obtain all possible values with their corresponding probability distributions (i.e., confidence degrees), by which we clean the incomplete databases. Experimental results showed the efficiency, accuracy and precision of our methods.

Keywords: Missing data cleaning, Bayesian network, Probabilistic database, Gibbs sampling, Probabilistic inference.

1 Introduction

Data quality is often affected by data anomalies, e.g., missing data, which prevalently exist in real world databases. It is necessary to carry out data cleaning to guarantee the data quality [1]. Actually, data cleaning is one of the critical mechanisms for companies to realize the full business value of big data in helping meet the quality, performance and scalability goals [2]. Many methods were used to clean missing data upon the additional information. For example, missing data items can be filled in by choosing the values satisfying the given functional dependencies [3]. A correct value for each missing data item can be derived based on the given conditional functional dependencies [4]. The correct values for missing data items could be found given aggregate constraints [5] by deleting the tuples that do not satisfy the constraints.

Actually, it is also necessary to fill in the missing value when the additional information is not available. But the correct value for each missing data item is difficult to be obtained in this situation [1, 6]. In this paper, we address the problem of data cleaning without additional information by providing a set of possible values for each missing data item rather than only one correct value. Particularly, we desire to infer

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 348–359, 2013.

the probability distributions of all possible values for the missing data items and then fill in the missing values by these distributions.

In recent years, some methods have been proposed to predict the possible values for missing data based on the probabilistic model. For example, Mayfield [7] presented a framework to infer the missing values by capturing attributes dependencies with a relational dependency network. Stoyanovich [8] provided a framework to infer the probability distributions for missing data by learning a meta-rule semi-lattice (MRSL) model for each attribute from incomplete databases. These methods are efficient, but model templates are required for constructing the structure of the probabilistic model. Moreover, the MRSL model cannot represent the dependencies of all attributes from a global point of view and two inference methods are necessary: one for single attributes and another for multiple attributes separately.

Therefore, in order to clean the missing values for arbitrary attributes, a probabilistic model is necessary to represent the dependencies among all attributes. This means that to fulfill the cleaning of missing data, we will have to address the following two problems: (1) constructing a probabilistic model from the given incomplete databases; (2) providing an efficient inference mechanism to predict the probability distributions of all possible values for the missing data items for both single and multiple attributes.

It is well known Bayesian network (BN) is an effective framework of representing dependencies among random variables [9]. A BN is a directed acyclic graph (DAG) where nodes represent random variables and edges represent dependencies among random variables. Each variable in a BN is associated with a conditional probability table (CPT) to give the probability of each state given parent states. Comparing to the above-mentioned probabilistic model, the global, qualitative and quantitative dependencies among all attributes can be well represented by means of BNs. Furthermore, uncertainties can be inferred effectively by BN inference algorithms [9]. Thus, in this paper, we adopt BN as the underlying framework for representing dependencies among attributes. We learn the BN from the given incomplete database and derive the probability distributions for the missing data item by the BN inference algorithm.

To learn BNs from the incomplete database, called IBN, we extend the classical dependency-analysis based BN learning algorithm [10] by incorporating the inherence of databases with missing values. From the inherence of data cleaning, missing values will always take a very small proportion of the whole database. Thus, learned from the complete part of the given incomplete database, IBN can represent the dependencies or characteristics of the whole database basically, although the IBN does not include the items in the missing values. This makes the IBN be reasonably looked upon as the underlying model of probabilistic inferences for cleaning missing data. Comparing with the MRSL-based inferences, the probability distributions for single and multiple missing attributes can be derived universally by IBN inferences. Many algorithms for BN's exact inferences have been proposed [11], but these methods are of the exponential complexity, which are not efficient and suitable enough with respect to the BN-based inference especially over large scale BNs. Thus, based on Gibbs sampling [11], we proposed an approximate inference algorithm to obtain the probability distribution based on the IBN.

For each missing data item (i.e., incomplete tuple) t in databases, its probability distribution derived by the IBN's inference is a set of all possible combinations of values of the attributes missing in t. The sum of all the probabilities of the filled

values is 1, which means that the distributions can be calculated in a principled fashion. It is known that probabilistic databases have been proposed to manage a probability distribution on a set of possible worlds [12, 13]. Exactly, we store the probability distributions for the missing data items in a probabilistic database.

Generally speaking, our main contributions can be summarized as follows:

- We propose an efficient dependency analysis method to learn the IBN from incomplete databases, as the basis for cleaning missing values.
- We propose an approximate inference method to predict the probability distributions of possible values for the missing data items, and correspondingly give an algorithm to clean the missing data.
- We implement the proposed algorithms and make preliminary experiments to test the feasibility of our method.

The remainder of this paper is organized as follows: In Section 2, we learn a BN from incomplete databases. In Section 3, we infer the probability distributions of missing data and then clean the missing data. In Section 4, we show experimental results. In Section 5, we conclude and discuss future work.

2 Learning Bayesian Network from Incomplete Databases

Learning a BN from databases always has two steps: first constructing the structure of BN and then learning the CPTs [9], where the former is critical and challenging.

A BN is a DAG $G=(V, E)$, where V is the set of nodes and E is the set of edges. If two nodes in a BN are conditionally independent, there will be no edge between them, and information theory based measures can be used to detect conditional independencies of nodes [10]. The mutual information of two nodes X, Y is defined as

$$I(X,Y)=\sum_{i=1}^{I}\sum_{j=1}^{J}P(x_i,y_j)\log_2\frac{P(x_i,y_j)}{P(x_i)P(y_j)} \tag{1}$$

The conditional mutual information is defined as

$$I(X,Y\mid Z)=\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}P(x_i,y_j,z_k)\log_2\frac{P(x_i y_j\mid z_k)}{P(x_i\mid z_k)P(y_j\mid z_k)} \tag{2}$$

That $I(X, Y)$ is smaller than a certain threshold ε means that X and Y are marginally independent. Given condition Z, that $I(X, Y| Z)$ is smaller than ε means that X and Y are conditionally independent.

Cheng [10] proposed a dependency analysis method to construct the structure of BN from complete databases based on the information theory. In this classical algorithm, a node ordering is given to specify a causal or temporal order of the nodes of a BN. Considering the characteristics of the incomplete database, we modify the above distribution for node X (or X and Y), the tuples with missing values on X (or on X and Y) will not be taken into consideration. Following, we illustrate this by an example.

Example 1. Table 1 shows a part of incomplete table containing 4 non-key attributes, where "?" indicates a missing value. For convenience, we denote *tub*, *smo*, *can*, *xray* by *t*, *s*, *c* and *x* respectively. When computing $P(t, s)$, tuple $t5$ should be not taken into consideration, $P(t=absent, s=smoker)=0.5$, $P(t=absent, s=nonsmoker)=0.25$ and $P(t=present, s=nonsmoker)=0.25$. $t5$ should be taken into consideration when computing $P(t, x)$ since the values on t and x exist.

Table 1. A part of an incomplete database table

id	tub	smo	can	xray
t1	absent	nonsmoker	absent	normal
t2	absent	smoker	absent	normal
t3	present	nonsmoker	present	abnormal
t4	absent	smoker	absent	normal
t5	absent	?	?	normal

Algorithm 1 describes the steps of constructing an IBN.

Algorithm 1. IBN-Construction

Input: T, an incomplete table; O, a vector of node

Output: $G=(V,E)$, a DAG of the IBN

Variables: L, a list of pairs of nodes and each $l \in L$ is a pair (v_i, v_j), $v_i, v_j \in V$ and $i \neq j$; Z, a cut set

Steps:

$V \leftarrow$ Set-Node-Ordering(O), $E \leftarrow \{\}$, $L \leftarrow \{\}$
if $I(v_i, v_j) > \varepsilon$ then $L \leftarrow L \cup \{(v_i, v_j)\}$ //By Equation (1)
sort L by the decreasing order of $I(v_i, v_j)$
$l \leftarrow L[0]$, $E \leftarrow E \cup \{l\}$, $L \leftarrow L-\{l\}$
for each $l(v_i, v_j)$ in L do
  if no open path[1] between (v_i, v_j) then
    $E \leftarrow E \cup \{l\}$, $L \leftarrow L-\{l\}$
  end if
end for
for each $l(v_i, v_j)$ in L do
  $Z \leftarrow$ Find-Cut-Set(v_i, v_j)
  if $I(v_i, v_j | Z) > \varepsilon$ then $E \leftarrow E \cup \{l\}$ //By Equation (2)
end for
for each e in E do
  if there are paths besides e between v_i and v_j then
   $E \leftarrow E-\{e\}$, $Z \leftarrow$ Find-Cut-Set(v_i, v_j)
   if $I(v_i, v_j | Z) > \varepsilon$ then $E \leftarrow E-\{e\}$
end for
return G

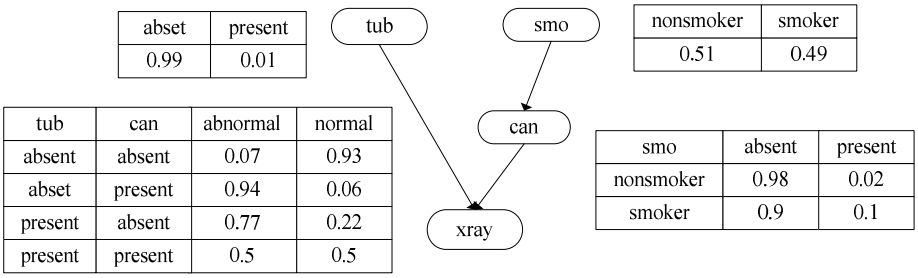
[1] A path that does not include head-to-head nodes is call open path.

For an incomplete database with n attributes, the conditional independency tests will be done for $O(n^2)$ times. Example 2 illustrates the execution of Algorithm 1.

Likelihood estimation [11] is commonly used to estimate the parameters of a statistical method by counting the frequency from tuples. We adopt this method to compute the CPT for each variable easily upon the IBN structure, where the probability is

$$P(x \mid y) = \frac{P(x, y)}{P(y)} \approx \frac{\text{the number of } (x, y)}{\text{the number of } y} \tag{3}$$

Example 2. For the incomplete table shown in Table 1, suppose the node ordering is {*tub*, *smo*, *can*, *xray*}. The mutual information of all pair of nodes obtained from databases are: $I(t, s)$=0.0, $I(t, c)$=0.0001, $I(t, x)$=0.033, $I(s, c)$=0.0278, $I(s, x)$=0.0139 and $I(c, x)$=0.192. Suppose ε is 0.01, so $I(c, x)>I(t, x)>I(s, c)>I(s, x)>\varepsilon$. So, L is {(c, x), (t, x), (s, c), (s, x)} and three edges (c, x), (t, x) and (s, c) can be added into E. Edge (s, x) will not be added since $I(s, x \mid c)= -0.0001$ is smaller than ε, where c is the cut set of s, x. Upon the obtained IBN structure, the CPTs can be computed based on Equation (3). Finally we can obtain the IBN as shown in Fig 1.



abset	present
0.99	0.01

nonsmoker	smoker
0.51	0.49

tub	can	abnormal	normal
absent	absent	0.07	0.93
abset	present	0.94	0.06
present	absent	0.77	0.22
present	present	0.5	0.5

smo	absent	present
nonsmoker	0.98	0.02
smoker	0.9	0.1

Fig. 1. An IBN learned from the incomplete table shown in Table 1

3 Cleaning Missing Data Based on Probabilistic Inferences

For each incomplete tuple t, we are to infer the probability distributions for the possible values of the attribute on which t's value is missing. All the possible values with probability distributions exactly constitute the representation of x-relation for uncertain data tuples, interpreted in [14]. An x-relation consists of one or more x-tuples independent of each other, each of which is a multiset of one or more mutually exclusive tuples, called alternatives. We then store the probability distribution of possible values for each missing data item as an x-tuple with a foreign key linked to the original incomplete table. Finally, we can fill in the missing value using the most probable one in corresponding x-tuples (i.e., the alternative with the largest probability).

It is known that Gibbs sampling is a Markov chain Monte Carlo algorithm and it always generates a Markov chain of samples. Thus, Gibbs sampling is particularly well-adapted to sampling the posterior distributions of a BN [11]. To infer the probability distributions of all possible values, we give Algorithm 2 as an approximate method for IBN inferences. First, we give the basic ideas as follows:

(1) We set a value to each attribute with missing value (i.e., nonevidence variable) and constitute the initial state including the attributes with complete values (i.e., evidence variables).

(2) We sample one of the attributes with missing values randomly and determine the value of the selected attribute from the conditional distribution under the current state. The new state can also be used for the next time of sampling. We repeat the sampling until the given threshold number of samples is reached.

(3) We get a set of samples, containing possible combination of values of the attributes with missing values, and those of the attributes with complete values. Then, the corresponding probability distributions can be achieved.

Algorithm 2. X-Tuple-Deriving

Input: X, missing attributes, $\{X_1, X_2, \ldots, X_n\}$; e, non missing attributes, $\{e_1, e_2, \ldots, e_m\}$; *ibn*, an IBN

Output: an x-tuple

Variables: $T[s]$, a vector of counts over s, initially zero; s, the current state of *ibn*, initially copied from e; x, a vector of values of X in s; $B[]$, a set of probability conditioned on the Markov blanket[2] of X, denoted as $MB(X)$; $s_{(-i)}$, the set $(X_1, X_2, \ldots, X_{i-1}, X_{i+1}, \ldots, X_w, e_1, e_2, \ldots, e_m)$; m, threshold of total number of samples to be generated

Steps:

```
x ← random values of X_i in X, s ← x∪e
for j←1 to m do
  if T[s] contains s then
    T[s] ← T[s]+1
  else
    insert s into T[s], T[s] ← 1
  end if
  select a query variable X_i from X randomly
  B[0] ← P(X=x_1|MB(X_i))   // X_i∈{x_1,x_2,…,x_k}
  for i←1 to k do
    B[i] ← B[i-1]+P(X=x_i|MB(X_i))
  end for
  generate a random value r∈[0,B[k]]
  X_i ← x_j where r <=B[j] //Determine the value of X_i
  s ← (s_(-i),X_i)
end for
T[s] ← T[s]/m
return T[s]
```

[2] A Markov blanket of X is the set of nodes composed of X's parents, its children and its children's other parents.

For an IBN with n nodes, the computations of probabilities conditioned on the Markov blanket are less than $O(nm)$ times. Following, we illustrate the execution of Algorithm 2 by an example.

Example 3. For the incomplete tuple $t5$ in Table 1, we are to obtain the probability distribution $P(s, c|t=absent, x=normal)$. We initialize the state s_0={s=*nonsmoker*, c=*absent*, t=*absent*, x=*normal*} and set 1 as the value of $N[s_0]$. Then, we randomly select s as a query variable and generate a value $r \in [0,1.0]$ randomly. Suppose r=0.67, and then we set s=*smoker* and generate a new state s_1={s=*smoker*, c=*absent*, t=*absent*, x=*normal*}. This procedure will be repeated for m times. Finally, we obtain the estimation (i.e., an x-tuple) of $t5$ shown in the right of Table 2.

Then, we can fill in the missing values by creating a new table including the x-tuple derived by Algorithm 2. For each incomplete tuple t in T, we call the function X-Tuple-Deriving in Algorithm 3 to generate an x-tuple t'. Then, we can select the alternative from T' with the largest probability and then update t in T.

Algorithm 3. Missing-Data-Cleaning

Input: T, an incomplete table; *ibn*, an IBN learned from the incomplete table T
Output: T', an x-relation corresponding to T
Variables: X, a set of attributes; e, a set of values of attributes; t', an x-tuple
Steps:

```
create table T'
for each incomplete tuple t∈T do
  //Attributes with missing values as query variables
  X ← Missing-Attributes(t)
  e ← Non-Missing-Attribute-Values(t)
  t'← X-Tuple-Deriving(X, e, ibn)
  insert t' into T'
  update t by the alternative with the largest probabil-
   ity in T' //t is the incomplete tuple in T
end for
return T'
```

Example 4. Revisiting the incomplete tuples in Table 1, we obtain the x-tuple for $t5$ by using Algorithm 2 and store it in Table 2, from which we select tuple $t5.1$ with the largest probability and update $t5$.

Table 2. x-tuple for $t5$ in Table 1

id	tub	smo	can	xray	prob
t5.1	absent	nonsmoker	absent	normal	0.564
t5.2	absent	smoker	absent	normal	0.430
t5.3	absent	smoker	present	normal	0.004
t5.4	absent	nonsmoker	present	normal	0.002

4 Experimental Results

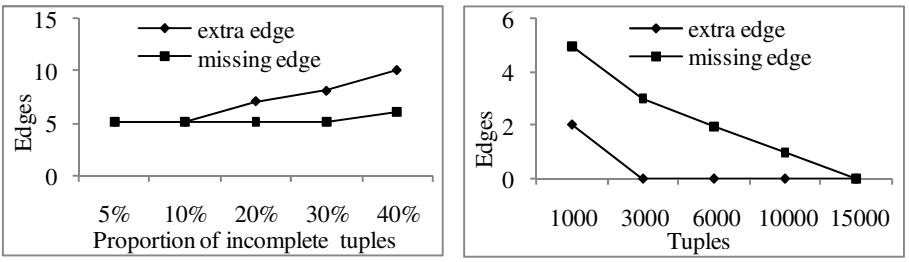
To verify the feasibility of the methods proposed in this paper, we implemented the presented algorithms. We mainly tested the accuracy and efficiency of IBN learning, and tested the convergence of the method of IBN inferences, and finally we tested the precision and efficiency of IBN-based missing data cleaning.

4.1 Experiment Setup

In the experiments, we adopted four classical BNs, Cancer Neapolitan (CN), Chest Clinic (CC), Car Diagnosis2 (CD) and Alarm (AL), widely used benchmarks. For each BN, we generated five original data sets of 1000, 3000, 6000, 10000 and 15000 tuples according to their probability distributions from Norsys [15]. We then generated five test data sets for each original one with 5%, 10%, 20%, 30% and 40% of incomplete tuples respectively, by setting one or more attributes to be NULL randomly. All the data sets were stored in MS SQL Server 2008 and all the codes were written in C#. The machine configurations are as follows: AMD Athlon64 X2 5000+ CPU, 2GB of main memory, running Windows 7 Ultimate 32-bit operating system.

4.2 Accuracy and Efficiency of IBN Learning

The edges which exist in the IBN structure but not exist in the true structure obtained from Norsys [15] are called extra edges. Edges which do not exist in the learned structure but exist in the true structure are called missing edges. We evaluated the accuracy of Algorithm 1 by recording the extra edges and missing edges. The extra edges and missing edges with the increase of incomplete tuples are shown in Fig. 2 (a). It can be seen clearly that the less the incomplete tuples, the more the learned IBN will be close to the true one. Meanwhile, there are few extra edges and missing edges in the IBN when the proportion of the incomplete tuples is about 5%. The extra edges and missing edges with the increase of tuples in the data set are shown in Fig. 2 (b). It is clear that the more the tuples in the given data set, the more accurate the learned IBN will be, which is consistent with the general conclusion for BN learning. Thus,



(a) Edges of Alarm BN with the increase of incomplete tuples

(b) Edges of Alarm BN with the increase of total tuples

Fig. 2. Accuracy of Algorithm 1 for constructing IBNs

Table 3. Accuracy of Algorithm 1 on various BNs

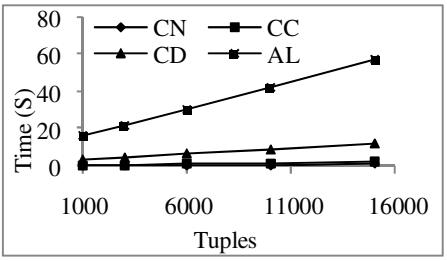
	extra edges	missing edges	correct edges
Caner Neapolitan	0	0	5
Chest Clinic	0	1	7
Car Diagnosis2	0	0	20
Alarm	0	3	43

we can conclude that Algorithm 1 is accurate for IBN learning. Further, for 15000 tuples with 10% incomplete tuples, we recorded the extra, missing and correct edges all the benchmark BNs, shown in Table 3.

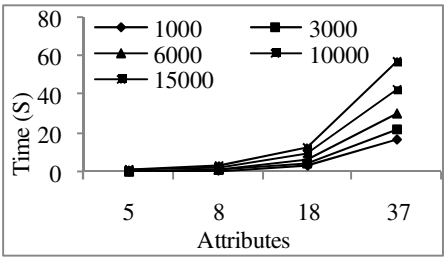
Then, Fig. 3 presents the execution time of IBN learning, including the time of DAG constructing and that of CPT computation. It can be seen that execution time is increased linearly with the increase of data tuples and nearly quadratically with the increase of attributes. This means that the execution time is not sensitive to the scale of the data set. In particular, we can also see that the cost of IBN learning is mainly dependent of the number attributes of the data set, as the node numbers of IBN, instead of that of tuples. Thus, our method for IBN leaning is efficient.

4.3 Convergence of the Inference Algorithm

It is pointed out that the posterior probabilities predicted by an approximate algorithm for BN's inferences are correct only if the sampling results are converged to a



(a) Execution time of IBN learning with the increase of tuples



(b) Execution time of IBN learning with the increase of attributes

Fig. 3. Execution time of learning IBNs with 10% incomplete tuples

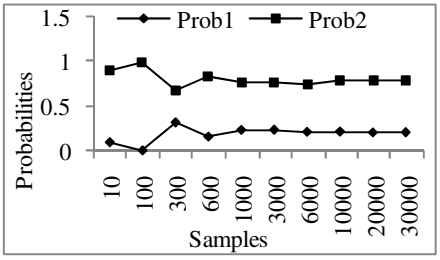


Fig. 4. Convergence of Algorithm 2

certain probabilistic value [11]. Thus, we tested the convergence of Algorithm 2 by recording the results upon the Cancer Neapolitan IBN under *Serum_Calcium= increased*, *Brain_Tumor=absent*, *Coma=present* and *Severe_Headaches=present*. Prob1 and Prob 2 in Fig. 4 is the probability of *Metastatic_Cancer = absent* and that of *Metastatic_Cancer=present* under the above evidences. It can be seen that Prob1 and Prob2 are stable around 0.2 and 0.8 respectively with the increase of the generated samples. The results show that the probabilities returned by Algorithm 2 converge to a certain value efficiently with just about 1000 samples, which guarantees the efficiency and correctness of Algorithm 2.

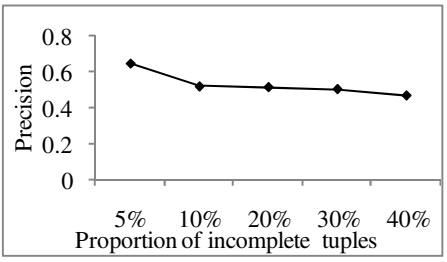
4.4 Precision and Efficiency of Data Cleaning

First, we compared the most possible value predicted by Algorithm 3 with the true value in the original data set. We used 1 (and 0) to denote the case that the predicted value is (not) the same with the true one. We defined the average precision by the mean of 0 or 1 for all incomplete tuples. Fig. 5 (a) shows the average precision of the most possible values obtained by Algorithm 3 on 1000 tuples, from which we can see that the precision will be decreased slowly with the increase of the proportion of incomplete tuples.

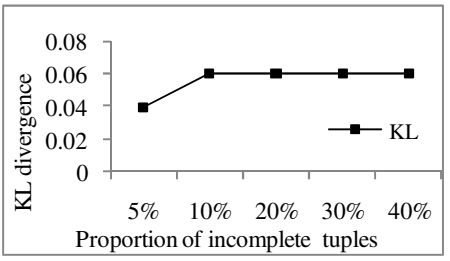
Meanwhile, we know that the measure of Kullback-Leibler (KL) divergence [16] is close to zero when the distributions are close to the results of Enumeration algorithm, as the exact BN inference algorithm [11]. The KL divergence of distribution Q from distribution P is computed by

$$D_{KL}(P \| Q) = \sum_i \ln(\frac{P(i)}{Q(i)})P(i) \tag{4}$$

Then, we recorded the KL divergence values for different proportions of incomplete tuples in the data set, shown in Fig. 5 (b). It can be seen that less the incomplete tuples, the smaller the KL divergence, i.e. the closer the two probability distributions, will be. This means that derived probability distributions are quite close to those obtained by the exact inference algorithm.



(a) Precision of Algorithm 3 with the increase of incomplete tuples

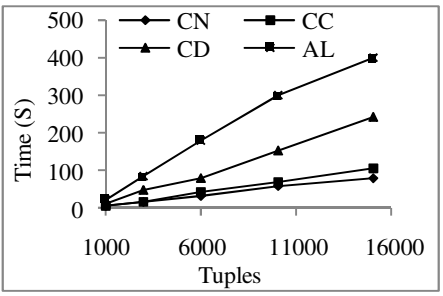


(b) KL divergence of Algorithm 3 with the increase of incomplete tuples

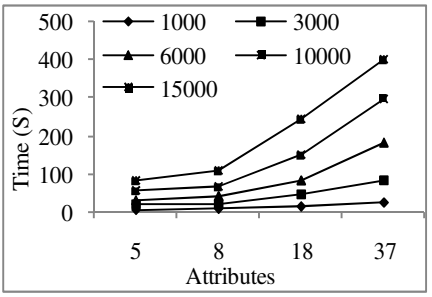
Fig. 5. Precision of Algorithm 3 under 1000 tuples

Table 4. Precision of Algorithm 3 on various numbers of tuples

tuples	CN		CC		CD		AL	
	precision	KL	precision	KL	precision	KL	precision	KL
1000	0.82	0.07	0.65	0.04	0.82	0.06	0.68	0.16
3000	0.74	0.11	0.55	0.05	0.75	0.06	0.66	0.21
6000	0.75	0.11	0.52	0.06	0.71	0.12	0.80	0.15
10000	0.76	0.10	0.49	0.06	0.72	0.10	0.80	0.14
15000	0.72	0.11	0.52	0.07	0.69	0.14	0.83	0.14



(a) Execution time as a function of the number of tuples.



(b) Execution time as a function of the number of attributes.

Fig. 6. Execution time of data cleaning with 5% incomplete tuples

Further, we recorded the results of cleaning on various numbers of tuples with 5% incomplete parts, shown as Table 4. Thus, the accuracy of Algorithm 3 for missing data cleaning is mainly determined by the IBN with a certain portion of incomplete tuples. From the perspective of real applications of data cleaning on an original data table with a small proportion of incomplete tuples, our method can work effectively.

Following, we recorded the execution time of Algorithm 3 shown in Fig. 6 for four BNs, each of which corresponds to five test data sets with 5% incomplete tuples. It can be seen that the execution time is increased linearly with the increase of tuples and attributes of the data sets, which verifies the efficiency of Algorithm 3.

5 Conclusion and Future Work

In this paper, we proposed the BN-based method for cleaning missing data. Focusing on the associations between attributes, we gave the methods for BN learning and inferences taking the given incomplete database as input. Theoretical and experimental analysis results verify the feasibility of our method.

To test our method further, we will make experiments on arbitrarily distributed data sets. As well, more efficient method for model learning and inferences will be considered by incorporating some optimization strategies. To extending our method to the realistic big data sets by incorporating the techniques of uncertain databases. Based on the methods proposed in this paper, we can explore the BN-based cleaning for redundant or wrong values. These are exactly our future work.

Acknowledgement. This paper was supported by the National Natural Science Foundation of China (61063009, 61163003, 61232002), the Ph. D Programs Foundation of Ministry of Education of China (20105301120001), the Yunnan Provincial Foundation for Leaders of Disciplines in Science and Technology (2012HB004), the Natural Science Foundation of Yunnan Province (2011FZ013), and the Foundation for Key Program of Department of Education of Yunnan Province (2011Z015).

References

1. Muller, H., Freytag, J.C.: Problems, Methods, and Challenges in Comprehensive Data Cleansing. Technical report, Humboldt-Universitat zu Berlin (2003)
2. Arasu, A., Chaudhuri, S., Chen, Z., Ganjam, K., et al.: Experiences with using Data Cleaning Technology for Bing Services. IEEE Data Engineering Bulletin, 14–23 (2012)
3. Beskales, G., Ilyas, I.F., Golab, L.: Sampling the repairs of functional dependency violations under hard constraints. PVLDB 3(1), 197–207 (2010)
4. Bohannon, P., Fan, W., Geerts, F., Jia, X., Kementsietsidis, A.: Conditional Functional Dependencies for Data Cleaning. In: Chirkova, R., Dogac, A., Ozsu, M.T., Sellis, T.K. (eds.) Proc. of ICDE 2007, Istanbul, Turkey, pp. 746–755. IEEE Computer Society (2007)
5. Chen, H., Ku, W.S., Wang, H.: Cleansing Uncertain Databases Leveraging Aggregate Constraints. In: Workshops Proc. of ICDE 2010, California, USA, pp. 128–135. IEEE Computer Society (2010)
6. Srivastava, D.: Analyzing Data Quality Using Data Auditor. In: Chen, L., Tang, C., Yang, J., Gao, Y. (eds.) WAIM 2010. LNCS, vol. 6184, pp. 1–1. Springer, Heidelberg (2010)
7. Mayfield, C., Neville, J., Prabhakar, S.: ERACER: A Database Approach for Statistical Inference and Data Cleaning. In: Elmagarmid, A.K., Agrawal, D. (eds.) Proc. of SIGMOD 2010, Indiana, USA, pp. 75–86. ACM (2010)
8. Stoyanovich, J., Davidson, S., Milo, T., Tannen, V.: Deriving Probabilistic Databases with Inference Ensembles. In: Abiteboul, S., Bohm, K., Koch, C., Tan, K.L. (eds.) Proc. of ICDE 2011, Hannover, Germany, pp. 303–314. IEEE Computer Society (2011)
9. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press (2009)
10. Cheng, J., Greiner, R., Bell, D., Liu, W.: Learning Bayesian Networks from Data: An Efficient Approach Based on Information Theory. Artificial Intelligence 137(1-2), 43–90 (2002)
11. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall (2009)
12. Cavallo, R., Pittarelli, M.: The Theory of Probabilistic Databases. In: Stocker, P.M., Kent, W., Hammersley, P. (eds.) Proc. of VLDB 1987, Brighton, England, pp. 71–81. Morgan Kaufmann (1987)
13. Huang, J., Antova, L., Koch, C., Olteanu, D.: MayBMS: A Probabilistic Databases Management System. In: Cetintemel, U., Zdonik, S.B., Kossmann, D., Tatbul, N. (eds.) Proc. of SIGMOD 2009, Rhode Island, USA, pp. 1071–1074. ACM (2009)
14. Benjelloun, O., Sarma, A., Halevy, A., Widom, J.: ULDBs: Databases with Uncertainty and Lineage. In: Dayal, U., Whang, K.Y., Lomet, D.B., Alonso, G.A., Lohman, G.M., Kersten, M.L., Cha, S.K., Kim, Y.K. (eds.) Proc. of VLDB 2006, Seoul, Korea, pp. 953–964. Morgan Kaufmann (2006)
15. Norsys Software Corporation, `http://www.norsys.com/`
16. Cover, T., Thomas, J.: Elements of Information Theory. Wiley and Sons (2006)

Stock Cloud Computing Platform:Architecture and Prototype Systems

Jianfeng Wu[1], Jing Sun[2], Xue Bai[2], Li Xue[2], Lili Lin[1], Xiongxiong Zhang[1], and Shuo Bai[1]

[1] Shanghai Stock Exchange, Shanghai 200120, P.R. China
[2] School of Computer Science, Fudan University, Shanghai 200433, P.R. China
{jfwu,lililin,xxzhang,sbai}@sse.com.cn
{jingsun,xuebai,xueli}@fudan.edu.cn

Abstract. The Challenge of Big Data leads to a great interests in Cloud Computing in both research and industry. Many Cloud Platforms have been proposed and implemented in various applications, however, there are few works focusing on designing integrated platforms from a data driven perspective for stock market. In this paper, we propose a cloud platform for stock market with four-tier architectures that introduces Infrastructure as a Service(IaaS),Data as a Service(DaaS),Platform as a Service(PaaS) and Software as a Service(SaaS). The proposed cloud platform integrates big data processing, data mining, and cloud computing technologies, and can provide high-performance computing and s data service as well. Finally, a case study on market surveillance in stock market validates the effectiveness and efficiency of the proposed prototype systems.

Keywords: big data, stock, cloud computing platform, architecture.

1 Introduction

Nowadays, a huge amount of data are generated and accumulated in scientific and business field. The big data raises many challenges for data analysis and knowledge discovery in all kinds of disciplines. For example,the large amount of data prohibit the applications of in-memory algorithms, and the needs for low-latency and real-time decisions are required in many industry scenarios, especially in stock market. Thus researcher are search for new solutions for the big data problem. Cloud computing [1,2,3]is one of the most important techniques that can meet this challenge. There has been some online applications for financial cloud computing. For example, NASDAQ provides the Market Replay[4] cloud service that acts as a replay and analysis tool and allows users to view the consolidated order book and trade data for NASDAQ, NYSE and other regional exchange-listed securities. NASDAQ Data-On-Demand[5] is another cloud service that provides queries, downloads, and analysis for history stock data. The NYSE Euronext's Community Platform is another financial cloud service that

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 360–370, 2013.

enable electronic trading, market data analysis, algorithms tests, regulatory reporting, and etc. Most of these financial cloud services are only based on historic data[6], and do not provide real-time services.

In this paper, we introduce a integrated cloud architecture for financial market composed of four tier, specifically, IAAS ,DAAS,PAAS and SAAS. To provide better analysis services, we integrate data mining techniques, statistical analysis, visualization management, parallel computing into our cloud platform and are capable of deal with Semi-structured/ unstructured data. The proposed cloud platform can provide various services including historic and real-time such as query services, market replay, model calculation, surveillance services, public opinion tracing, and etc. In addition, we also implement two prototype cloud computing systems for the financial market: stock cloud service and surveillance systems. We also make a detailed analysis of the prototype system and evaluate the performances of the market surveillance service.

The rest of the paper is organized as follows: Section 2 introduces the related work. Section 3 provides the details of the security cloud computing technical architecture. In Section 4,we will show two prototype systems of cloud service,evaluate performance and make corresponding analysis.

2 Related Work

The development of information technology(IT) greatly contributes to the progress of the society and science. Meanwhile all walks of life constantly informatization bring great challenges to information technology. With the wide usage of IT systems, huge amount of data is generated, especially in the fields of scientific computing and commercial computing. The stock industry is one of several key sectors in the financial field. Before establish a financial data center, data must be gathered.Stock field holds a great majority of financial data resources ,which provide a rich data basis for the construction of a finance data center. A variety of stock business have been built up many independent private IT systems to deal with. It is urgent to find a way to provide a high-performance computing capacity for the big data.

Various applications in stock industry require a lot of computing resources.The temporal distribution required by the computational resource of these applications is non-uniform.Framework based on cloud computing can dynamically organize various computing resources and realize transparent and scalable computing system architecture, so as to meet the various application computing need.This framework also improves the efficiency and rationality of computing resources.Just as stock industry is developing quickly and the business is ever growing, it is the demand for the ability of computing and data services.Traditional IT systems are hard to meet the requirements of high reliability, high timeliness and high security of security business, leading to frequent trading halt. For example, on February 25, 2011, London Stock Exchange system encountered failure which led to transaction interruption for about 4 hours. On May 18, 2012, Facebook run into technical problems in the NASDAQ.These stock trading accidents

not only cause huge losses to investors, but also hurt the reputation reputation of the stock exchange.Cloud computing provide new solutions to resolve similar problems in the stock industry.

Cloud computing is a new business computing model of sharing infrastructure resources with the quickly development of computing, storage and communication technology. It is different from traditional calculation with the center of local personal computer,while it is internet-centric, by constructing one or more data center connected with a large number of ordinary machines and network devices (one million),and then store the data to the data center.It provides a convenient and transparent data storage and computing services for the upper layer services and applications. In order to extend applications, we should concern the hardware infrastructure and the cost of equipment running.With the spread of cloud computing technology, it appears to affect various technical fields [7,8].Typical cloud computing products are Amazon Web Services (AWS) [9],Application software engine (Google AppEngine, GAE) [10], Microsoft's software as a service (SaaS) products and Windows Azure cloud operating system [11].Beside the above commercial cloud,the open source cloud is also growing fastly.For example, Hadoop[12],Eucalyptus[13,14],OpenStack[15] and so on. Although the global stock industry have a huge demand for cloud computing, only NASDAQ,NYSE(New York Stock Exchange)[16], Euronext and CME(Chicago Mercantile Exchange) officially launched cloud computing services [17]. As an emerging IT technology, cloud computing has been studying by many stock exchange institutions.

3 System Architecture

3.1 Stock Cloud Computing Technical Architecture

By studying on the existing open source cloud computing technologies, we select the scene-demand cloud technologies(such as real-time data processing) and then we build the corresponding cloud architecture according to the framework of each functional requirements characteristics, mainly in the view of demand of data processing capacity and computing power. The stock cloud computing architecture is divided into four tier: infrastructure resource management, stock data processing engine, stock supporting platform, stock of internal and external service platform. The four-tier are corresponding to the Iaas,DaaS,PaaS and SaaS.The architecture is shown in Figure 1.

The lowest layer is infrastructure resource management. It provides hardware support and system software support for other services.On the top of infrastructure resource management is stock data processing engine layer. It provides structured and unstructured stock data acquisition, conversion, cleaning, loading, storage, and fast processing. Stock supporting platform provides data mining platform,standard business process. It contains three parts:application support platform, data service support platform and distributed computing support platform. On the top of the architecture is internal and external service

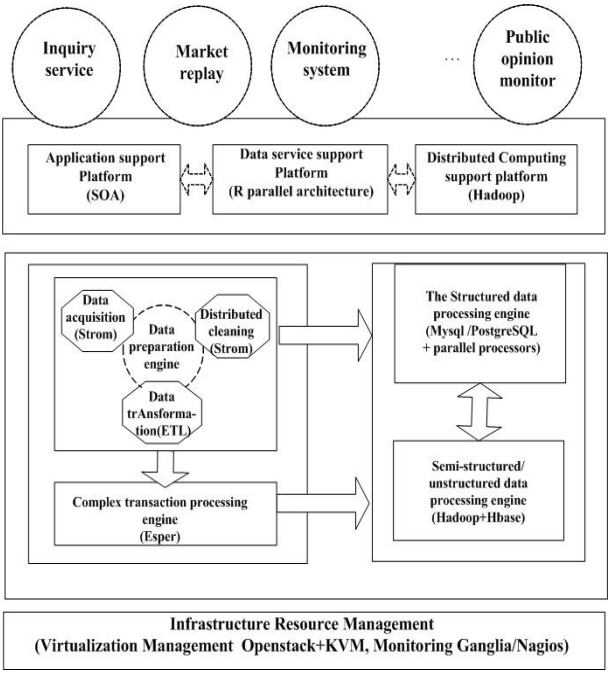


Fig. 1. The basic technical architecture of stock cloud

platform. It contains inquiry service,market replay,monitoring system, public opinion monitor and so on.

3.2 Infrastructure Resource Management

Infrastructure resource management is primarily responsible for the management and maintenance of infrastructure resources. Local infrastructures are connected through the network. Physical machines is virtualized. And then we build the distributed storage service. It also can use the agent to view the information of infrastructure resources,realize remote control and construct virtualized resource pool through virtualization technology.

The core functions include resource supply, storage management, template library management, network management, and resource scheduling. Main function of resource supply is providing the task queue. Storage management mainly manages the life cycle of the storage management mirror, as well as private storage and public storage. The template library management provides a common template, user templates, and incremental template. Network management mainly provides DHCP service, DNS service, and Vlan service. Resource scheduling mainly provides functions of VM placement and VM consolidation.As an virtualization management platform, the prime functions of OpenStack are virtualization and management of virtualization resources. We use Openstack as

the primary implementation technique of IaaS(infrastructure as a service). OpenStack is a free software and open source project authorized by the Apache license.It is an IaaS software that anyone can create their own cloud computing services on it.

3.3 Stock Data Processing Engine

Stock data processing engine provides massive amounts of structured and unstructured stock data acquisition, conversion, cleaning, loading, storage, and fast processing ability.The vast majority of business data resource such as transaction data and reporting data in the internal of stock exchange are structured data. They are mainly stored and managed in relational database system, mainly due to that the traditional relational database can take advantage of the SQL language character in efficiently query operations for structured data. However, with the growing data, the relational database system could hardly meet the query efficiency of the stock business requirements.Therefore, we select the parallel database technology as the processing engine of structured data.

Unstructured data is another important part of the resources of the stock data, including text data, chart data and other types. These unstructured data record a large number of stock information. Unstructured data mining analysis has significance for studying stock markets. For example, by analyzing the Stock and text messages in the network,we can predict stock market fluctuations and analysis of the associations of stock market event. For lacking well-defined data format, it is greater difficult to deal with unstructured data than structured data.

The cloud computing platform constructs semi-structured data and unstructured data processing engine using Hadoop. HBase realizes column-based data storage. HDFS supports distributed and unstructured data storage. The semi-structured and unstructured stock data can be unified storage and be managed in Hadoop platform. By the MapReduce mechanism, the cloud platform realizes data processing parallelization and solves the problem of computationally intensive tasks demanding for computing power.Meanwhile it takes advantage of the distribution of data storage and maximize realize computing parallelization. Cloud computing platform use two processing modes of batch and online respectively to handle the parallel business and structured data computing under static and real-time scene.

3.4 Data Mining Platform

Data mining platform mainly provides a support for mining in big data. Since the calculation scene is distributed parallel computing, the technical difficulties mainly lie in the parallelization of data mining technologies.

Stock cloud computing selects R and Hadoop as the main data mining platform technologies. R language is a popular data analysis tools.It has many advantages in data statistics, as well as in data mining. R language combines with Hadoop can realize effective method of data parallel processing in the cloud

environment. It uses distributed file system HDFS and distributed computing model MAPReduce to realize R parallel framework.R and Hadoop platform contain three components: (1) R data analysis is a driver program which provides a data analysis environment; (2) Hadoop cluster provides host data and runs Jaql and some R sub-process. Hadoop offers a range of computing nodes which are used to store data and perform calculations. Jaql provide a high-level language for Hadoop data processing and query and editing a series of MapReduce tasks. (3) R-Jaql bridge provides communication and data conversion mechanism for the connection of more than two components. It handles the tasks: executing Jaql query, converting the query results to R form, dispersing R process to each different Hadoop node. R and Hadoop Parallel architecture is shown in Figure 2.

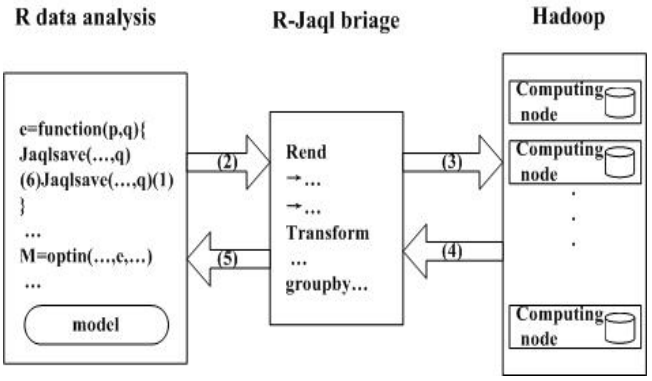


Fig. 2. R and Hadoop Parallel architecture

The corresponding process is as following: (1)computing gradient by query; (2)sending the query parameter into Jaql; (3)executing query on the parallel computing node; (4)returning the result; (5)converting the result into R; (6)using the result of R.

4 Cloud Service Prototype System

From many stock cloud computing for business applications,we select two representative applications to create cloud application prototype systems.They are the stock data services cloud prototype and surveillance system cloud prototype.

4.1 Stock Data Services Cloud Prototype

Based on the cloud business demand, the technology architecture prototype is shown in Figure 3.

For deploying stock cloud service prototype,we need the following support service: 1. Structed data and unstructed data processing service.They mainly are

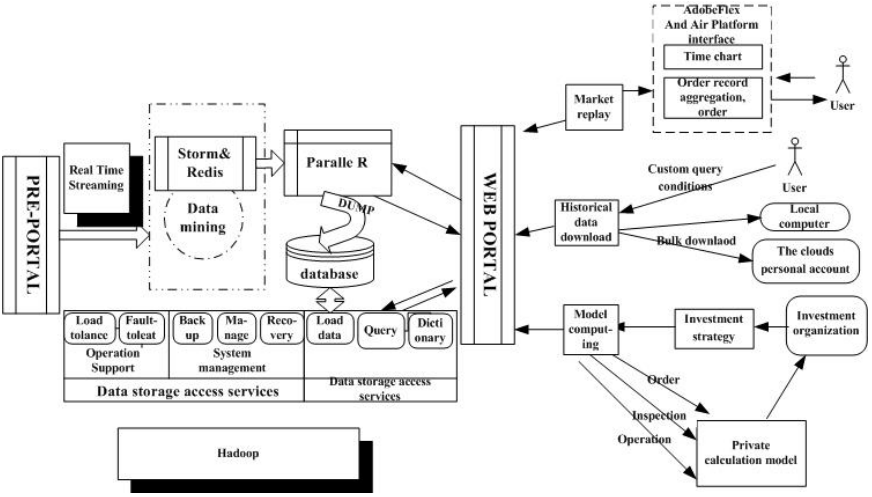


Fig. 3. Stockdata services cloud prototype

parallel R service and MapReduce parallel data processing service. 2. Real-time data processing support service.It mainly designs real-time data stream processing service which developed by Storm and Redis. Stock cloud data service provides the following application service: 1.Market Replay. It takes Amazon Simple Storage Service for historical market data persistence. A basic function of Market Replay program is to provide real market activities replaying, the effect of which is same to what a transaction person saw in real-time trading.In order to be able to give any time order records, sorting and aggregation abilities for order records information updating is essential. 2. Historical data downloading. It provides historical stock transactions data and other available public data on-demand downloading service. Institutional investors can customize query conditions, find the customized historical data what they need, bulk download in the local computer or in the personal cloud account. 3. Model calculations. Different investment institutions customize its private computing model according to its own unique investment strategy. And then they take these security models as the important foundation of the security investment. This business mainly provides the institutional investors model of customizing, testing and computing.

4.2 Design and Implementation of Market Surveillance System Prototype

Design of Cloud Market Surveillance Prototype System. Market surveillance system prototype is shown in Figure 4. It takes parallel database as data storage and query software

Loading Market Surveillance System Data. The main contents of the surveillance system include parallel database loading, parallel database query and parallel database communication. Data loading process is as follows: 1.Loading request is sent to the data partitioning module by fault tolerance and load balancing module; 2.The data partitioning module send data to the object allocation module according to the distribution strategy ; 3.Object allocation module select a data loading modules by database connection pool perform loading to database. Loading data flow is shown in Figure 5.

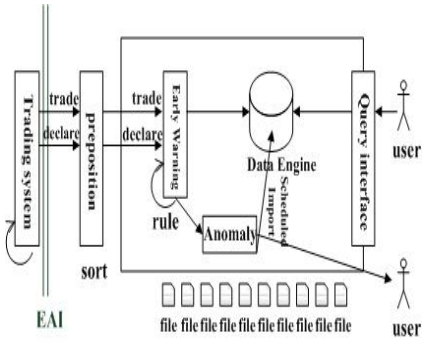


Fig. 4. Cloud market surveillance prototype system

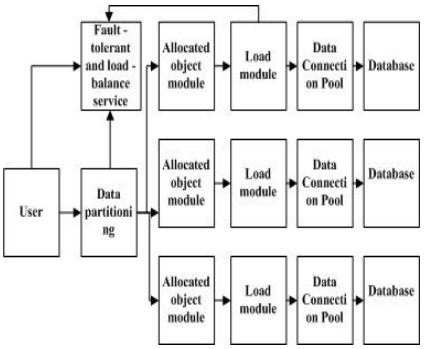


Fig. 5. Loading market surveillance system data

Data Communication Strategy of Market Surveillance System. The concrete realization of the communication data between nodes completed by MPI.NET. It can realize MPI programming design of object-oriented style by MPI.NET. In working condition, each of the data nodes has a dedicated MPI service thread which meets the demand between data nodes and the transmit-receive data. The concrete realization of the data nodes and management nodes communication are completed by System.Net classes based on.Net technology. In working condition, the data service nodes monitor in the specified port and establish a queue for access requests received.If an access request comes, inserting the request into the access requests queue.At the same time generating and launching a new thread to handle request. Then the data service nodes continue to monitor in the port. The processing result of each access request returns to original link. Therefore, the end of the data service node may be started simultaneously multiple threads to handle access requests from different management nodes arriving simultaneously, which can be done independently of each other.

Data Distribution Strategy of the Market Surveillance System. Hash method is based on a property value of the tuple through Hash function assigning tuple to the specified processing node. It can make the data distribution uniformity by designing a well function. Hash method not only can effectively support

the large amount of data access operations, but also can effectively support data manipulation with a low selectivity predicate in the division of property. However, for the selection predicate is a range of queries on the division of properties, Hash method will use all processors to complete the scope of the query, which increases the system overhead. Hash method is shown in Figure 6.

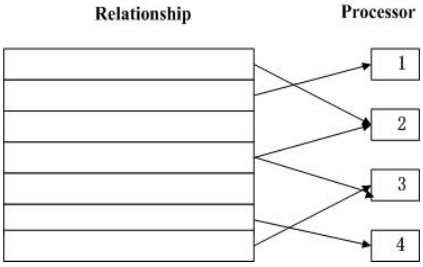


Fig. 6. Hash method

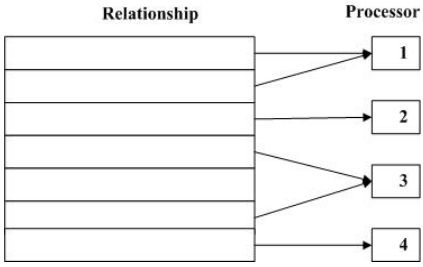


Fig. 7. Range method

Range Method. Range method is based on the range of the property of relation R.It divides the relationship into a collection of N tuples and then assigns them to the N processing nodes. Range method not only can effectively support access operations on the large dataset and on the division of properties with low selectivity predicate. It is also very suitable for range query in the division of properties. According to the query conditions, it can fast access directly to the data of meeting the conditions, which eliminates the cost for communication among many processing nodes and saves the system overhead. Range method is shown in Figure 7.

Query Optimization Strategies of the Market Surveillance System. At present, most attention has been focused on the semantic query optimization method based on Agent technology. The method uses Agent technology in Artificial Intelligence to achieve optimizing of parallel database query. It uses multi-agent technology automatically to find the integrity constraints for a given query, and then modifies the given query to more effectively equivalent query. It greatly improves the efficiency of connection operation between multi-relationship, so as to reduce the joint operation and shorten query time optimization. It realizes the Agent-based semantic query optimization.

Experiment and Result Analysis. A prototype system of market surveillance system with 7 desktop computers, named VM01 to VM07. The hardware configuration of the computer is: Intel I5 3.2GHz CPU, 8GB RAM, 1TB HD, 1GB RJ-45 NIC. The operating system is RedHat Enterprise Linux server. We use Hadoop, HDFS and HBase to set up system.

Query response time is the evaluation criterion for testing stock data.Most queries are simple queries(the number of queries involving records is less than 500). Some other queries are complex one.

case1 : Query trading volume detail for one user.case2: Query trading volume detail for 1000 users.

The query time of case1 and case2 are listed in the following Table 1:

Table 1. The query time in different cases

case	case1			case2		
Records number (million)	20	40	80	20	40	80
1PC(s)	0.406	0.699	0.912	2.013	3.621	5.066
7PC Cloud(s)	0.078	0.126	0.188	0.408	0.746	1.021

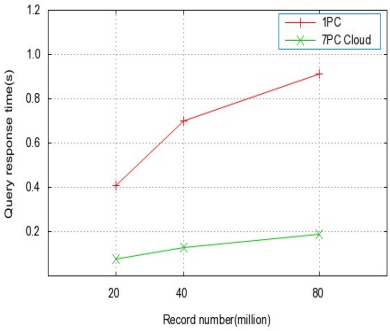


Fig. 8. One user query

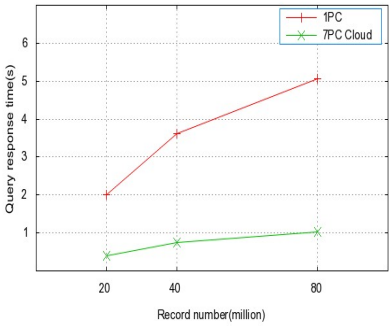


Fig. 9. 1000 users query

The experiment results are shown in Figure 8 and Figure 9. Comparing the response time of queries, we found that the larger the total amount of data, the more advantages of cloud computing appears. For using range method of distributed computing strategy, computing tasks are more evenly assigned to each PC. It greatly improves the efficiency of distributed computing. When the data volume is little, cloud computing platform performance is lower than a single PC . The task scheduling between multiple PCs yields network communication overhead. In addition, we can see that the query response time greatly increase when the number of users is large. Because the larger number of users, the larger query result sets and more complex queries. Links between tables overhead are large. The result is affected by the factors of the IO and task scheduling.

5 Conclusion

Traditional IT system is difficult to meet the stock business demand of high reliability, high timeliness and high security.It is urgent to need the high-performance of computing services and data services.In this paper, we design a stock cloud computing platform. We also give two prototype systems with their designing implementation and evaluation.The experiment results show the superiority of our stock

cloud computing platform.And this platform gives a useful reference for the construction of cloud computing platform.The importance of stock issues in the cloud environment presents a gradual upward trend. Designing safe operation of the system becomes an important issue in the stock cloud field.In the future,privacy protection of stock cloud need to be studied,so as to solve the problems of data stock and privacy data stock and privacy protection brought by the cloud service computing model, virtualization management style and operational mode.

Acknowledgment. This work is supported in part by the National science and technology Pillar Program. (NO.2012BAH13F02 and NO.2012BAH13F04).

References

1. Youseff, L., Butrico, M., Da Silva, D.: Toward a Unified Ontology of Cloud Computing. In: Grid Computing Environments Workshop, pp. 1–10 (2008)
2. Chen, K., Zheng, W.M.: Cloud computing:System instances and current research. Journal of Software 20(5), 1337–1348 (2009)
3. Raj, H., Nathuji, R., Singh, A.: England p. Resource management for isolation enhanced cloud services. In: Sion, R. (ed.) Proc. of the 2009 ACM Workshop on Cloud Computing Security,CCSW 2009,Co-Located with the 16th ACM Computer and Communications Security Conf., CCS 2009, pp. 77–84. Association for Computing Machinery, New York (2009)
4. Essvale Corporation Limited, Bizle Professional Series,Business Knowledge For It. In: Trading And Exchanges
5. http://www.cloudbulls.com/cloud-strategy-for-financial-markets-and-exchanges
6. York, J.: Banking Technology Forecast is Partly Cloudy(2010), http://www.cloudbulls.com/banking-technologyforecast-is-partly-cloudy.2010.8
7. Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D.J., Silberschatz, A., Rasin, A.: HadoopDB:An architectural hybrid of MapReduce and DBMS technologies for analytical workloads. PVLDB, 922–933 (2009)
8. Ahrens, M., Alonso, G.: Relational databases,virtualization,and the cloud. In: Abiteboul, S., Böhm, K., Koch, C., Tan, K.L. (eds.) Proc. of the 27th Int'l Conf. on Data Engineering (ICDE 2011), p. 1254. IEEE Computer Society Press, New York (2011)
9. Garfinkel, S.L.: An evaluation of Amazon's grid computing services: EC2,S3 and SQS, TR-08-07. Harvard University, Cambridge (2007)
10. GAE, http://code.google.com/intl/en/appengine/
11. Microsoft, http://www.microsoft.com/windowsazure/
12. Hadoop, http://hadoop.apache.org/
13. Eucalyptus project (EB /OL) (December 26, 2008), http://eucalyptuscs.ucsb.edu
14. Nurmid, Wolskir, Grzegdrczykc, et al.: The Eucalyptus Open-source cloud computing system. In: Proc of Workshop on Cloud Computing and its Applications (2008)
15. Openstack, http://www.openstack.org/
16. NYSE Euronext, http://nysetechnologies.nyx.com/hosted-solutions/community-platform
17. NASDAQ, http://www.cloudave.com/11981/nasdaq-omx-serves-big-historical-stock-data-from-the-cloud

A Novel Method for Identifying Optimal Number of Clusters with Marginal Differential Entropy

Bo Shu[1], Wei Chen[2,⋆], Zhendong Niu[1], Changmin Zhang[1], and Xiaotian Jiang[1]

[1] School of Computer Science and Technology,
Beijing Institute of Technology, Beijing, 100081, China
[2] Agricultural Information Institute,
Chinese Academy of Agricultural Sciences, Beijing, 100081, China
{shu.bo.china,chanmionezhang}@gmail.com, chenwei@caas.cn,
zniu@bit.edu.cn, 0532yangya@163.com

Abstract. Clustering evaluation plays an important role in clustering algorithms. Most of recent approaches about clustering that evaluate and identify the optimal number of clusters need to calculate the distances between data points pair-wisely or evaluate the entropy in the entire dimension space and have high computational complexity. In this paper, we propose an entropy-based clustering evaluation method for identifying the optimal number of clusters which first projects the clusters centroids to each of its individual dimensions, then accumulates the marginal differential entropy in each dimension. With the sum of marginal entropies we can analyze the performance and identify the optimal number of clusters. This method can dramatically reduce the computational complexity without losing accuracy. Experiment results show that the proposed method has high stability under various situations and can apply to massive high-dimensional data points.

Keywords: Clustering Evaluation, Information Theory, Differential Entropy.

1 Introduction

Clustering algorithms are wildly applied in a lot of domains, such as: data mining, machine learning [1], pattern recognition [2] [3], business intelligence [4], image analysis [5], information retrieval [6] [7], bioinformatics [8], etc. The evaluation of clustering results, which is also called cluster evaluation, plays an important role in verifying the effectiveness of the algorithms.

Typically, clustering evaluation is divided into two categories: internal evaluation and external evaluation. The internal evaluation is based on the similarity of intra-clusters and the difference of inter-clusters. Current common internal

⋆ Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 371–382, 2013.

evaluation methods include Davies Bouldin Index [9], Dunn index [10], Silhouette Validation Method [11], etc. For example, Davies Bouldin Index is defined as (1):

$$\begin{aligned} S_{i,q} &= \left(\frac{1}{|A_i|} \sum_{\underline{x} \in A_i} \|\underline{x} - \underline{v}_i\|_2^q \right)^{\frac{1}{q}} \\ d_{(ij,t)} &= \left\{ \sum_{s=1}^{p} |v_{si} - v_{sj}|^t \right\}^{\frac{1}{t}} = \left\| \underline{v}_i - \underline{v}_j \right\|_t \\ R_{i,qt} &= max_{j \in c, j \neq i} \left\{ \frac{S_{i,q} + S_{j,q}}{d_{ij,t}} \right\} \end{aligned} \tag{1}$$

where $\underline{v}_i$ is the i-th cluster centroid; q and t are integers larger than 1, they are independent of each other; $|A_i|$ is the number of points in cluster A_i; $S_{i,q}$ is the mean distance from each data point in the i-th cluster to the centroid of this cluster; $d_{(ij,t)}$ is the distance between the centers of the i-th cluster and the j-th cluster, where a small value means the better clustering performance.

From above we can see the main idea of internal evaluation: the more similar the elements within a cluster are, the more dissimilar the elements between clusters are, the better performance the clustering algorithm has. If we use distance to express the similarity between data points, then this observation can be put in another way: the smaller the distance of the elements within a cluster are, the larger the distance of the elements between clusters are, the better performance the clustering algorithm has.

The external evaluation approach includes Rand Index, F-measure, Jaccard index, etc.

The Rand Index is defined in (2):

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \tag{2}$$

where a, b, c, d is defined respectively as follows:

a, the number of pairs of data points that are in the same set with clustering algorithm and in the same set in pre-labeled cluster scheme.

b, the number of pairs of data points that are in the different set with clustering algorithm and in the different set in pre-labeled cluster scheme.

c, the number of pairs of data points that are in the same set with clustering algorithm and in the different set in pre-labeled cluster scheme.

d, the number of pairs of data points that are in the different set with clustering algorithm and in the same set in pre-labeled cluster scheme.

The range of Rand Index is $[0, 1]$ and a larger value means the better performance of clustering algorithms.

From these descriptions, it can be concluded that the main idea of external evaluation is to compare the clustering scheme with pre-label classified scheme. The more similar the clustering scheme and pre-labeled classified scheme are,

the better performance the clustering algorithm has. In practical there is no pre-labeled data and we can only adopt the internal evaluation.

When evaluating the result of clustering, we can often find that for a certain clustering algorithm the clustering performance is a strongly associated with the number of clusters. So currently some evaluation methods analyze the relation between some measures of clustering result and the number of clusters. These methods can also help clustering algorithm identify the optimal number of clusters, such as [12] and [13].

Robert Tibshirani, Guenther Walther, Trevor Hastie [12] use W_k to measure the compactness within clusters. W_k is defined as (3):

$$\begin{aligned} D_r &= \sum_{i,i' \in C_r} d_{i,i'} \\ W_k &= \sum_{r=1}^{k} \frac{1}{2n_r} D_r \end{aligned} \tag{3}$$

where $d_{i,i'}$ is the distance between two points in cluster C_r, n_r is the number of points in the r-th cluster. The clustering scheme has k clusters. For each data sample, this method first creates a null reference distribution data sample having the same number of points with the data sample, then clusters the reference sample as well as the data sample, and finally calculate the $Gap_n(k)$. $Gap_n(k)$ is defined as (4):

$$Gap_n(k) = E_n^*(log(W_k)) - log(W_k) \tag{4}$$

where E_n^* denotes the expectation under a sample of size n, and the k when $Gap_n(k)$ gets its maximum value is the optimal number of clusters. Also, we can see the optimal number of clusters is located at an elbow point where the reduction speed of W_k suddenly drops. However, this method need to pair-wisely calculate the distances between data points within each cluster, and it is impractical when dealing with massive high-dimensional data.

In [13], Keke Chen and Ling Liu present a method based on information entropy to evaluate the effect of clustering. They first investigate the entropy property of the categorical data, then cluster the data with their hierarchical HierEntro method, at last they evaluate the result with the BkPlot method and determine the best number of clusters by finding the peaks/valleys at the second-order differential of the curve. However, this method needs to calculate the information entropy of data sample, and it is also impractical when dealing with massive high-dimensional data.

Inspired by [12] and [13], in this paper we present a novel method based on information theory. Instead of calculating information entropy of the data sample in the entire space, we use the sum of marginal entropies in each dimension to evaluate the performance of clustering, which can dramatically reduce the complexity of algorithm. First we project the centroids of clusters to each dimension and calculate the single dimension marginal information entropy, then we accumulate the marginal entropies in each dimension. From the sum of the marginal

entropies of each dimension, we can evaluate the performance of clustering result and find the optimal number of clusters.

The rest of this paper is organized as follows. We propose our algorithm and explain the deriving process with a concrete case in section 2. In section 3, we test our conclusions with different synthetic datasets, compare our method with the other two existing evaluation measures and do some analysis. At last we draw the conclusion in section 4.

2 Algorithm

We first cluster the data sample with different number of clusters, then for each clustering result we calculate the sum of single dimensional marginal differential entropy. By observing the change trend of the sum of single dimensional marginal differential entropy, we can draw some general rules on finding the optimal number of clusters. The details of this algorithm will be depicted below.

2.1 Notations and Problem Statement

Let $A = \{A_1, A_2, \cdots, A_d\}$ be a set of continuous real number domains. S is a d-dimensional hyperspace, $S = A_1 \times A_2 \times ... \times A_d$. The data sample consist of a set of d-dimension points, $V = \{v_1, v_2, \cdots, v_n\}$,$v_j = \{v_{j_1}, v_{j_2}, \cdots, v_{j_d}\}$. The i-th component of v_j is an element in A_i. The number of different value of i-dimension is u_i, the different values of i-dimensional component in V are $v_{i1}, \cdots, v_{iu_i}$, the distances between different adjacent values in i-dimension are $l_{i1}, \cdots, l_{iu_i}$, $l_{ix} = v_{i(x+1)} - v_{ix}$. When $j + 1 = 1$, consider $l_{ij} = \infty$; when $j = u_i$, consider $l_{i(j+1)} = \infty$, and the number of data points in i-dimension with the value $v_{i1}, \cdots, v_{iu_i}$ is $k_{i1}, \cdots, k_{iu_i}$, so,

$$\sum_{x_1=1}^{u_1} k_{1x_1} = \cdots = \sum_{x_i=1}^{u_2} k_{ix_i} = \cdots = \sum_{x_d=1}^{u_d} k_{dx_d} = n \tag{5}$$

We will observe the change trend of the sum of the single dimensional marginal differential entropy in the data sample, along with the relationship between number of clusters and other evaluation measures.

2.2 Calculate Single Dimensional Marginal Differential Entropy of Data Sample

According to theorem of Glivenko-Cantelli, When $n \to \infty$, the empirical distribution function $F_n(x)$ converge to cumulative distribution function $F(x)$ with probability 1, so here we treat $F_n(x)$ as $F(x)$. So for the i-th dimension, the probability distribution function and probability density function are:

$$F_i(x) = \frac{\sum_{j=1}^{w} k_{ij}}{n} \left(v_{iw} \le x < v_{i(w+1)}\right) \tag{6}$$

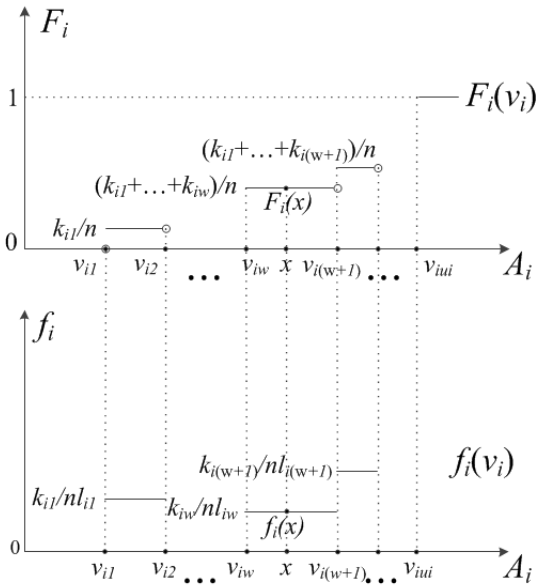


Fig. 1. Probability distribution function and probability density function of a single dimension dataset

$$f_i(x) = \frac{k_{iw}}{nl_{iw}} \left(v_{iw} \leq x < v_{i(w+1)}\right) \tag{7}$$

Formula (8) calculates H_i , the marginal distribution differential entropy of the i-th dimension:

$$\begin{aligned} H_i = -\int_{-\infty}^{+\infty} f(x) \log f(x)\,\mathrm{d}x &= \sum_{j=1}^{u_i} \int_{v_{ij}}^{v_{i(j+1)}} \frac{k_{ij}}{nl_{ij}} log \frac{nl_{ij}}{k_{ij}} \mathrm{d}x \\ &= \sum_{j=1}^{u_i} \frac{k_{ij}}{n} log \frac{nl_{ij}}{k_{ij}} = \frac{1}{n} \sum_{j=1}^{u_i} k_{ij} log \frac{nl_{ij}}{k_{ij}} \end{aligned} \tag{8}$$

When $j = u_i$, as we know, the entropy of variable in a bounded interval is also a bounded real number. As $n \to \infty$, the expectation of contribution of each point to differential entropy tends to 0, so here we define $k_{iu_i} log \frac{nl_{iu_i}}{k_{iu_i}} = 0$.

2.3 Evaluation Method

The evaluation process can be divided into four steps:

1 Cluster data sample with different number of clusters from 1 to K, $K \leqslant n$.
2 For each clustering scheme, replace each data point with its cluster centroid.
3 Calculate the sum of single dimension marginal entropies of each clustering result.
4 Find the optimal number of clusters with the conclusions depicted below.

Now we illustrate this method with a concrete case. Figure 2 represents a 2-dimension data sample with 100 points in 4 pre-set clusters. By using the k-means method, we obtained the clustering results with the number of clusters varying from 1 to 100. For each clustering result, we calculate W_k, Davies Bouldin index, and the sum of marginal entropies. The results are shown in Figure 3 (a), and the details when number of clusters is 4 is shown in Figure 3 (b).

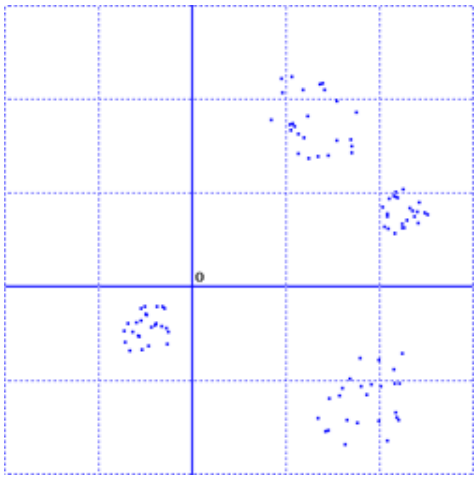


Fig. 2. Data sample with 100 points in 4 clusters

Figure 3 (a) shows the change trend of the sum of entropies, W_k, Davies Bouldin index along with the number of clusters. In order to display their relations clearly, we plot W_k with $W_k/300$ and Davies Bouldin index with 4 times of the original value in Figure 3 (a). From Figure 3 (a) we can see that the sum of entropies roughly keeps increasing while the speed (or in other word the first order derivative) keeps decreasing. At the positions of the pre-set number of clusters k or about $k * 2^n$, there are some convex area. W_k keeps declining and its first order derivative sharply increases at pre-set number of clusters. Davies Bouldin Index is roughly decreasing and reachs its local minima at pre-set number of clusters, then it increases and fluctuates irregular and finally reduces to zero when the number of clusters equals to the number of data points.

Figure 3 (b) shows the partial enlarged details of Figure 3 (a) near the pre-set number of clusters. We can see that when the number of clusters reaches 4, the reduction speed of W_k suddenly decreases. At the same time, a embow emerges. The curve of the sum of entropies has a convex, and Davies Bouldin Index has a local minimum.

After repeated experiments and observations with many datasets, we draw the following conclusions about identifying the optimal number of clusters with the sum of single dimension marginal en1tropies.

1 The change trend of the sum of the single dimension marginal entropies increases along with the number of clusters, while its first order derivative

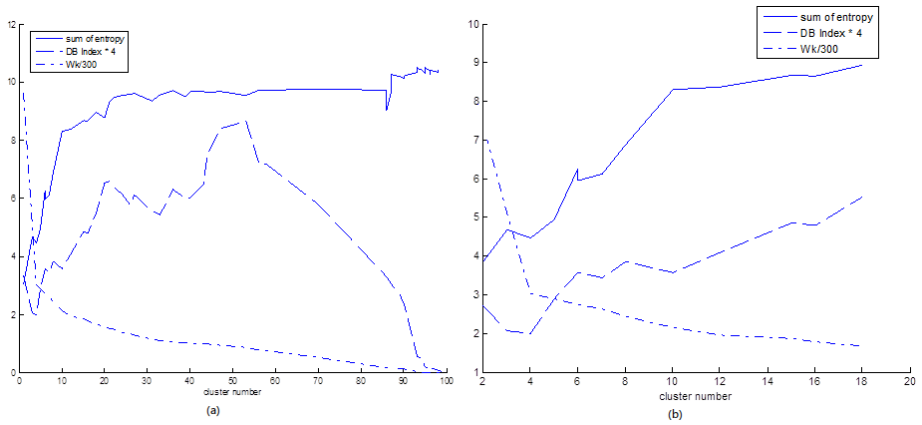


Fig. 3. Relationship between number of clusters and W_k, Davies Bouldin Index, and the sum of marginal entropy

gradually declines. When its first order derivative is continuously less than threshold δ, we consider the sum of entropies entering a "plain area". In the experiments of this paper, we set δ to 0.25. The minimum number of clusters in the plain area is located at the "elbow", and it has the best performance.

2 If there exists a convex at the left side of the "elbow", the smallest value in the convex may be the optimal number of clusters (there may exist several convex area).

3 If there is no convex at the left side of the "elbow", we use the minimum number of clusters in the plain area as the optimal number of clusters.

4 If there are several "elbow", we can only use other measures, such as W_k or Davies Bouldin Index, to determine the optimal number of clusters.

2.4 Complexity of the Method

Our method needs to calculate the sum of single dimension marginal differential entropies. First we project each point to d dimensions and its computation complexity is $O(nd)$. Then we sort the components of each point in each dimension and its computation complexity is $O(nlogn)$. Finally we calculate the marginal entropy in a dimension and the computation complexity of this part is $O(n)$. So the total computation complexity is $O(d(2n + nlogn))$. Comparing to other measures, this is comparatively low. For example, the computation complexities of W_k and Davies Bouldin Index are $O(dn^2)$.

3 Experiment Result

We use synthetic dataset to test our conclusions with increasing the number of dimensions or data points, and we also compare with the W_k method as baseline. The synthetic data generation method will be described in 3.1.

3.1 Generation of the Synthetic Data

We use the synthetic data to evaluate our algorithm. The generation program of the synthetic data can create two kinds of data distributions: uniform distribution and normal distribution. The uniform distribution generation method needs the following parameters: $\langle core_1, \cdots core_k \rangle$ the set of d-dimensional core points, $\langle c_{i1}, \cdots c_{id} \rangle$ the coordinate of $core_i$, n_i the number of data points generated by $core_i$, and $\langle r_{i1}, \cdots r_{id} \rangle$ the radius in each dimension of $core_i$. The method of the generated points is as follows: first we define all the above-mentioned parameters, then we choose the core points one by one and generate data points with the number of n_i. The coordinate of the j-th dimension of a data points generated by $core_i$ is c_{ij} added by a value randomly created in the interval of $\lfloor -r_{ij}, +r_{ij} \rfloor$. So the data points generated by a core point are all located in a hyper rectangle, the centroid of which is the core point. The number of parameters in the generation process

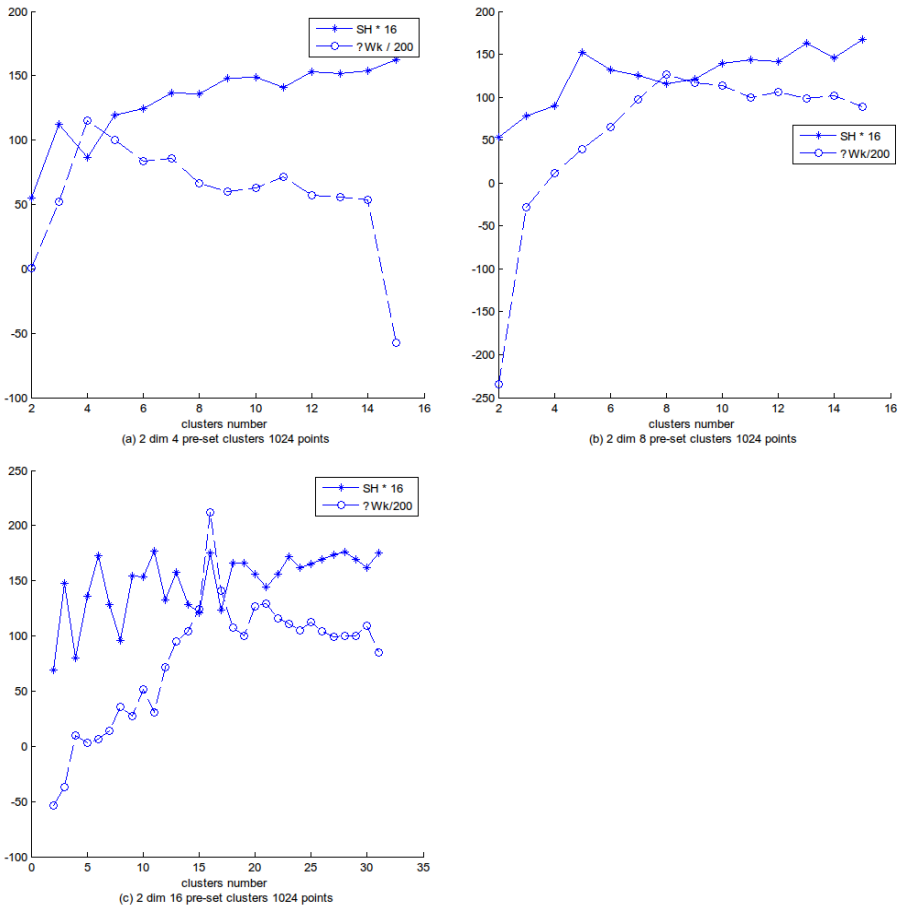


Fig. 4. Test with different number of pre-set clusters

of normal distribution dataset are similar to that of the uniform distribution, except that c_{ij} means μ_{ij}, r_{ij} means σ_{ij} in the j-th dimension of the i-th normal distribution component and n_i is the number of points in the i-th normal distribution component. We generated the coordinate of each dimension of each point with single dimension normal distribution model.

3.2 Synthetic Data Results

We test our conclusions with datasets that have different numbers of data points, pre-set clusters, and dimensions. We pre-set the number of data points, clusters, and dimensions, but randomly select generation types (uniform or normal distribution) to generate a data sample, then observe whether the data sample matches our conclusions.

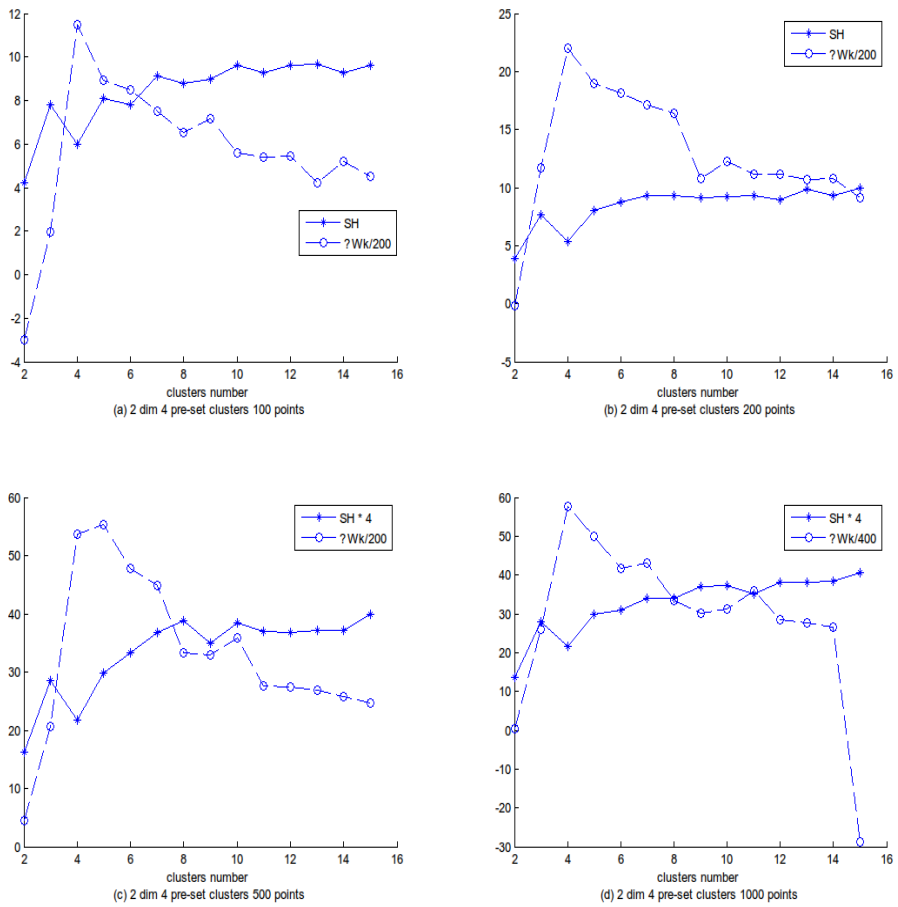


Fig. 5. Test with different number of data points

Figure 4 shows the relationship between the number of clusters and the sum of marginal entropy or W_k with different pre-set clusters number. We can see although there exists some fluctuations at the left side of pre-set clusters number, the convex of sum of entropies and the maximum $Gap_n\,(k)$ emerged at the point of the pre-set number of clusters and it matches our conclusions.

Figure 5 shows the relationship between the number of data points and the sum of marginal entropies or W_k with different numbers of data points. We can see that the convex of the sum of entropies and the maximum $Gap_n\,(k)$ emerged at the point of the pre-set number of clusters. The increasing number of data points has no influence on the convex position.

Figure 6 shows the relationship between the number of dimensions and the sum of marginal entropy or W_k with different number of dimensions. We can see the convex of sum of entropies and the maximum $Gap_n\,(k)$ emerged at the point

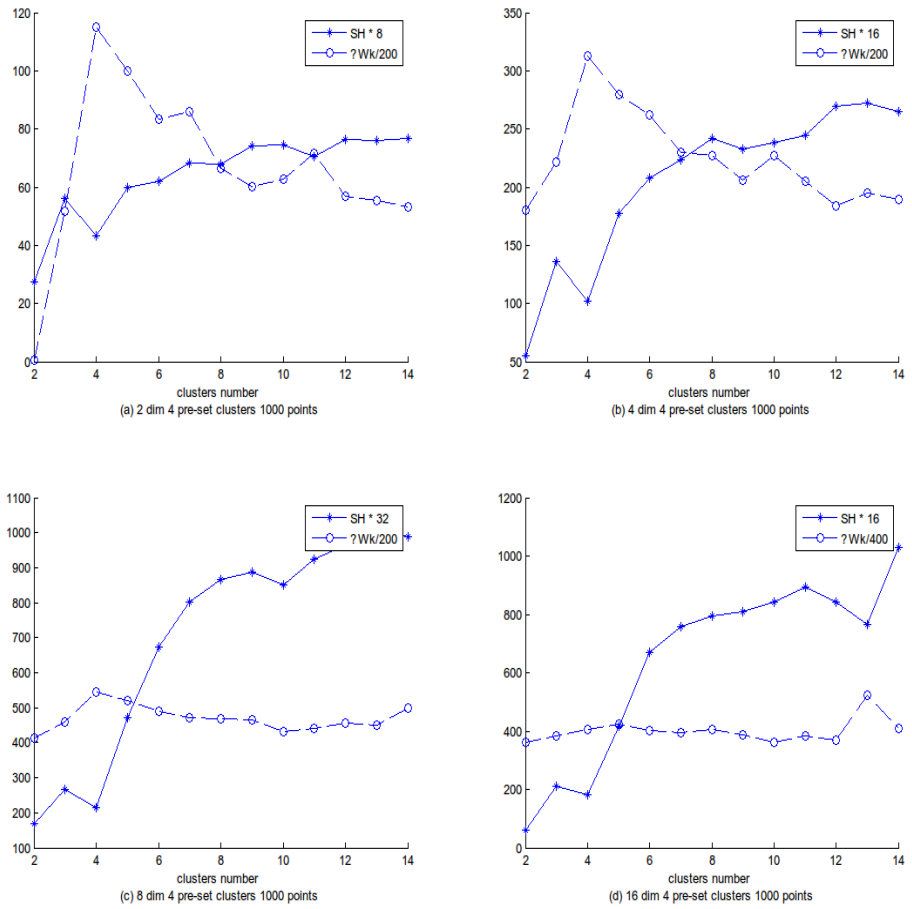


Fig. 6. Test with different number of dimension

of the pre-set number of clusters. The increasing number of dimensions has no influence on the convex position.

4 Conclusions

Most of the current research on clustering evaluation and the optimal number of clusters identification have high computation complexity. In this paper, we proposed an entropy-based clustering evaluation method for identifying the optimal number of clusters can dramatically reduce the computational complexity while keeping accuracy. The experimental result shows that our method has high stability when dealing with various situations. Moreover, it can apply to massive data points with high dimension. As to the future work, we will test our method with the clustering schemes which is generated with other clustering methods, such as DBSCAN [14], BIRCH [15], CLIQUE [16], etc. We will adopt more complicated metric to make our method more accuracy to locate the best clustering number.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (project no. 61003263 and no. 61250010), the National "973" Project of China (No. 2012CB720702), the CETV NewMedia Learning mall, the 111 project of Beijing Institute of Technology, Project of the Ministry of Agriculture of China "Agriculture Monitoring, Forecast and Informatization", and Key Projects of Natixonal Key Technology R&D Program during the Twelfth Five-Year Plan Period (No. 2012BAH20B04).

References

1. Wernick, M., Yang, Y., Brankov, J., Yourganov, G., Strother, S.: Machine learning in medical imaging. IEEE Signal Processing Magazine 27(4), 25–38 (2010)
2. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research 3, 1157–1182 (2003)
3. Foroutan, I., Sklansky, J.: Feature selection for automatic classification of non-gaussian data. IEEE Transactions on Systems, Man and Cybernetics 17(2), 187–198 (1987)
4. Wang, J., Wu, X., Zhang, C.: Support vector machines based on k-means clustering for real-time business intelligence systems. International Journal of Business Intelligence and Data Mining 1(1), 54–64 (2005)
5. Richards, J.A.: Remote sensing digital image analysis. Springer (2012)
6. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval, vol. 1. Cambridge University Press, Cambridge (2008)
7. Singhal, A.: Modern information retrieval: A brief overview. IEEE Data Engineering Bulletin 24(4), 35–43 (2001)
8. Li, W., Godzik, A.: Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics 22(13), 1658–1659 (2006)
9. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence (2), 224–227 (1979)

10. Dunn, J.C.: Well-separated clusters and optimal fuzzy partitions. Journal of Cybernetics 4(1), 95–104 (1974)
11. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53–65 (1987)
12. Tibshirani, R., Walther, G., Hastie, T.: Estimating the number of clusters in a data set via the gap statistic. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63(2), 411–423 (2001)
13. Chen, K., Liu, L.: The" best k" for entropy-based categorical data clustering. In: Proceedings of the 17th International Conference on Scientific and Statistical Database Management, pp. 253–262. Lawrence Berkeley Laboratory (2005)
14. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD (1996)
15. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. ACM SIGMOD Record 25, 103–114 (1996)
16. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications, vol. 27. ACM (1998)

Threshold Selection for Classification with Skewed Class Distribution

Xiaofeng He, Rong Zhang, and Aoying Zhou

Software Engineering Institute
East China Normal University, Shanghai, China 200062
{xfhe,rzhang,ayzhou}@sei.ecnu.edu.cn

Abstract. In document classification, threshold selection receives little attention, particularly in binary classification cases where threshold selection was largely ignored as a trivial task of a post-processing step. In webpage classification, however, we are facing a problem involving huge number of webpages usually with highly imbalanced class distribution. Due to the budget constraint, a reliable estimate of the threshold is required on only small size of human judged webpages. A good threshold selection criterion also need be adopted in highly imbalanced class distribution situation with positives being very spares in the sample set. These challenges make the threshold selection a non-trivial task for webpage classification. In this paper, we propose a novel cost efficient approach of threshold selection method for binary webpage classification with highly imbalanced class distribution. We construct a small sample set by applying stratified sampling on the webpages. The human judged samples are expanded to reflect the true class distribution of the webpage population. Experimental results show that false positive rate leads to more stable threshold estimate.

1 Introduction

Document classification is a very active research field in machine learning and data mining, especially in the Internet age when huge quantity of digital documents, the webpages, are readily available for information analysis. The webpage classification is an essential component of web search. The classification result is used from the very beginning of search activity, crawling the internet, to the final stage of the service, presenting ranked webpages to search engine users. Since no crawler can exhaust all Internet documents, to selectively crawl the webpages, the classifiers need predict high quality pages to crawl. Given the classification information, say spam or adult scores output by a classifier, decision will be made as whether to index the webpage or not. This information is also a very important feature for ranking function which presents users with highly relevant webpages. Many of these prediction tasks are of binary classification. The hard cutoff value, threshold, on the prediction score must be specified in order to assign class label to the webpages. This threshold is determined by threshold selection method.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 383–393, 2013.

Threshold selection method is an relatively under-investigated field in document classification field. It was regarded as a simple post-processing step after the classification models have been built applied to the documents. Current research on determining threshold mostly focuses on multi-class classifications. Thresholding on binary classification was normally either ignored as trivial task which simply chooses a cutoff value corresponding to pre-defined criteria such as precision for instance, or treated as another learning procedure which maps the prediction score to probabilistic score [7,13]. Probability thresholding proposed by Lewis [12] relies greatly on classifiers producing accurate probability of the class membership. For probabilistic classification model, however, all probability scores are proximately accurate because of the various factors affecting the quality of trained model. This inherent inaccuracy of the probabilistic model was verified by the works of Bennett, Ghani, and Yang [1,6,22] in the case of naive Bayesian classifiers.

In situation of webpage classification, often the class distribution of webpage data is highly imbalanced in that negatives are far more common than positives. The fraction of positives, or positive ratio is normally way below 20%, or even under 10%. To construct a validation set for threshold selection, simple random sampling of the webpages will likely result in the sample containing only a few positive points. Data sparsity results in high degree of uncertainty in estimated threshold values when they are based on selection criteria such as precision, recall, F_1 measure and their variants. The thresholds obtained on different samples generated by following same procedure can be quite different.

In this paper, we propose a stratified sampling of webpage data to construct the evaluation set for threshold selection for binary classification tasks, and found that false positive rate brings the most stable threshold selection results. Our major contributions are three folds. 1) We adapted the stratified sampling methods to construct the evaluation set, which was often ignored in Web document classification where the population size is huge. The boundaries of the strata were determined by both sampling error bound as well as the human judgment budget. 2) We proposed to use False Positive Rate as a stable criterion for threshold selection for binary classification. 3) We performed our experimental results on Web document to empirically verify our proposed approach.

The rest of the paper was organized as follows. We first briefly review existing thresholding methods in Section 2. Common sampling methods were explained in Section 3. Discussion of the criteria used as the metric for classifier performance measure was discussed in Section 4. In Section 5, we present an overall picture of our solution to address the threshold selection problem in context of webpage classification of highly imbalanced class distribution. We illustrate the experiment results of our approach in Section 6 using semi-synthetic data sets. We conclude this paper and discuss future works in Section 7.

2 Current Thresholding Methods

Threshold selection policies normally fall into two categories: *analytical* threshold selection and *experimental* threshold selection [15].

Analytical threshold selection determines the threshold theoretically by maximizing an effectiveness function [12]. It usually utilizes the probability of class membership generated by the classifiers, hence is also called *probability thresholding*. As indicated by Bennett, Yang independently [1,6,22], the probability of class membership by the classifiers such as Naive Bayesian classifier does not accurately represent the true degree of class membership of the documents. It tends to converge exponentially to either 0 or 1 as the number of features increases. Threshold determined on the probability naturally will far from optimal. Furthermore, in many cases, the theoretical analysis is not easily available. We have to choose the threshold experimentally.

Experimental policy determines threshold by optimizing some criteria on the validation sets. Three common strategies applied to multi-class classification are **RCut**, **PCut** and **Scut**[11,20].

These threshold tuning algorithms have been used in classification applications such as in [19]. Yang [20] proves that SCut is superior to PCut, while RCut is consistently inferior to other two strategies. Yang also proposed two alternative strategy for RCut and SCut, called **RTCut** and **SCutFBR** respectively. $RTCut$ reduces the harsh trade-off between precision and recall imposed in RCut. $SCutFBR$ is proposed in order to reduce the risk of overfitting inherent in SCut. Detailed of $RTCut$ and $SCutFBR$ can be found in [21].

3 Sampling Webpages for Threshold Selection

To estimate the threshold, we need construct a reliable validation set by sampling the document for editors to judge their class membership. In the case of webpage classification, there are three major characteristics we must bear in mind when sampling the webpages.

- highly imbalanced class distribution of the webpages
- daunting size of document population
- limited budget available for judgment

All these pose great challenge for threshold decision for web scale document classification.

From different classifier results, we notice that the positive ratio is very small, less than 10% in some classification applications. Most positives are concentrated on the tail of the score distribution. See Figure 1 for example where most positives have high scores. Simple random sampling in this case only gives a handful of positive samples even when we sample a considerable number of web pages. The sparsity of positive samples in the validation set unavoidably leads to unreliable estimation of the threshold.

Three questions must be addressed when we create validation set for threshold selection.

1. Which sampling strategy to use in order to obtain enough positive samples?
2. How large a sample is sufficient for threshold prediction?

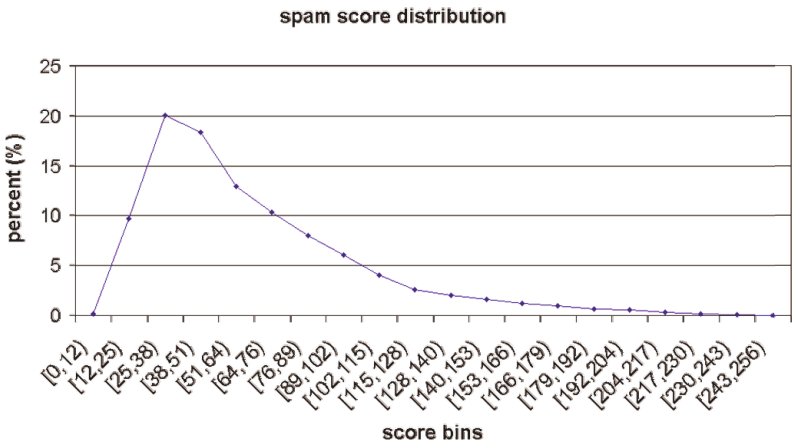


Fig. 1. Spam score distribution of 1M random webpages. The X-axis indicates the classification scores. 20 bins are generated with bin boundaries listed. Y-axis is percentage of the data in the bins.

3. How to transform the sample set so that the threshold or other metric obtained is a good estimate of the true value?

Before answering these questions, we want to review some sampling methods first. Sampling methods are studied extensively [18,10,2]. There are two major sampling strategies, non-probability sampling and probability sampling. non-probability sampling is a non-random sampling approach which includes

- Purposive sampling: choose samples with certain purpose in mind
- Quota sampling: allocate some quota to each type of population
- Convenience sampling: choose samples at the convenience of the researchers

where population is the entire data pool to be sampled from.

Probability sampling chooses the samples based on probability theorem. All data points in the population have non-zero probability of being sampled. Probability sampling includes:

- **Simple random sampling**: all data points have equal chance to be sampled independently.
- **Systematic random sampling**: arrange data into groups of ordered list, then sample at fixed position from each group.
- **Stratified random sampling**: population was split into strata (subgroups), then randomly sample from each stratum. It has two subtypes.
  - **Proportional stratified random sampling**: sample same percentage of data from each stratum.
  - **Disproportional stratified random sampling**: sample same number of data from each stratum.
- **Cluster sampling**: group population in a hierarchical fashion then randomly sample along this hierarchy.

To decide how to sample the population, we must factor in another constraint: only limited budget is available for editorial judgment, which limits the number of webpages to be judged, but several rounds of editorial judgment are typically needed in order to train a classifier with reasonable quality. Hence the final strategy should be carefully designed so as to gain maximum benefit from the human judged samples. Because of the highly imbalanced class distribution, we choose to use disproportional stratified sampling in order to obtain the adequate positive samples.

For the second question, since we do not have prior knowledge about the class distribution, power analysis for sample size estimation is not feasible. Instead, we resort to sampling margin of error to estimate the sample size for each stratum. Assume we randomly sample N data points. Let p_1 be the percentage of observed positives, and p_2 be percentage of observed negatives. The standard deviation is

$$std = \sqrt{p_1 p_2}.$$

This can be easily proved by assuming x_i to be random variables taking binary values: 1 for positive samples and 0 for negative samples. Further let n_+ be the number of positive samples and n_- be the number negative samples, then the mean value of x_i is $p_1 = n_+/N$.

$$\begin{aligned} std &= \sqrt{\frac{\sum_{i=1}^{N}(x_i - p_1)^2}{N}} \\ &= \sqrt{\frac{n_+(1-p_1)^2 + n_-(0-p_1)^2)}{N}} \\ &= \sqrt{p_1 p_2} \end{aligned}$$

The standard error of the percentage becomes

$$SE = \frac{std}{\sqrt{N}} = \frac{\sqrt{p_1 p_2}}{\sqrt{N}}. \tag{1}$$

Then the margin of error of sampling with 95% confidence interval can be calculated as

$$\text{margin of error} = 1.96 \times SE \tag{2}$$

$$= 1.96 \times \frac{\sqrt{p_1 p_2}}{\sqrt{N}} \tag{3}$$

Note that the margin of error is related to the sample size N, not the population size, which means if we obtain large amount of samples, we will have higher confidence on the final statistics. From the above calculation, we can get a rough idea on what margin of error of the sample will be for a given sample size. This is especially useful when we request minimal editorial budget.

To answer the third question, we must transform the judged sample data such that its class distribution reflects the real world distribution. There are two

ways to do it. One is to expand the sample set such that the data ratio reflects the ratio in the population. The other is to assign as weight the corresponding data ratio to each data point. We tested two sample expansion methods: simply duplicating the labeled samples to reach the required data ratio, and sampling from neighbors for those who have too few sample points. Experiments show that both methods lead to similar results. So we use the former method in the rest of the experiment.

4 Criteria for Threshold Selection

The classification result can be represented by a confusion matrix as in Table 1.

Popular metrics for classifier performance evaluations are accuracy, precision, recall and F_1 measure. Accuracy measure has bias towards dominant class, hence not suitable for the classification task involving imbalanced class distribution. Precision, recall and F_1 measure are defined as:

$$\text{Precision:} \quad p = \frac{TP}{TP+FP}; \quad \text{Recall:} \quad r = \frac{TP}{TP+FN}$$
$$F_1 \quad \text{measure} = \frac{2rp}{r+p}$$

Our purpose is to find an reliable estimate of optimal threshold. **Precision** is a natural candidate of threshold selection criteria. Precision as the criterion is widely used to evaluate the performance of the classifiers, as well as to determine the threshold. It is a successful choice for threshold selection when the class distribution of the sample set is close to true distribution, and if the positive points in the sample set are not so sparse. But in reality, we are doing random sampling of population with unknown distribution, and the sampling error can be regarded as a source of class distribution change. Furthermore, due to the varied performance of the classifiers, as well as the finite size of the sample set, we cannot guarantee that the positive ratio will be larger as the prediction score becomes larger. That is, precision is not monotonically increasing as the score increases, which will cause some problem when determining the threshold. We are sampling the webpages of imbalanced class distribution under constrained budget, so the positives in the sample set is like to be sparse. This also causes inaccurate estimate of the precision, which leads to considerable change in threshold prediction. Precision under such circumstances is not the best choice of the criterion any more.

Another criterion, **best F_1 measure**, turns out to be intractable in that the result is not easy to be fine-tuned. In many cases, it gives threshold corresponding to very low precision, though higher recall. This is undesirable since in the

Table 1. Confusion matrix of class distribution

	Predicted Class	
True Class	TP	FN
	FP	TN

application, we do not want to reject too many negatives. That means, we want to keep false positive rate low, even if the recall is relatively low compared to the threshold obtained by best F_1 measure, where false positive rate (*fpr*) is defined as

$$fpr = \frac{FP}{FP + TN}$$

We must emphasize that when the application puts strict requirement on the precision, then the precision should be used. Which criteria to use for threshold selection highly depends on the application the threshold selection method serves.

5 Threshold Selection Algorithm for Imbalanced Distribution

In the classification problem of highly imbalanced class distribution, we must devise a reliable sampling procedure which can give us a sample set easily being manipulated to reflect the real distribution of the population, while the human judgment is affordable. To achieve this goal, we take disproportional sampling approach to obtain webpages within our budget, then expand the judged sample in order to get a sample set which is more close to the real world distribution. The threshold selection criteria then can be applied to this expanded sample set for good threshold estimation.

We summarize our threshold selection algorithm in Algorithm 1.

Algorithm 1. thresholdSelection (population N, sampleSize M)

1. for population N, generate histogram of 20 bins of equal length;
2. merge bins into 3 or 4 strata based on criteria in Section 3;
3. $dataRatio[i] = n[i]/N$, where $n[i]$ is data size in stratum i;
4. estimate $sampleSize[i]$ using margin of error, where $\sum sampleSize[i] = M$;
5. random sampling from each stratum;
6. weight the sample points, or expand samples according to corresponding $dataRatio[i]$;
7. estimate threshold based on some criteria such as false positive rate;

6 Experimental Analysis

We performed experiments on a small size of human labeled data set. The classifier we build is gradient boosting tree. The classification scores then are converted from [0 1] range to [0 255] range so it can be stored in one byte. Note that this method can be applied to any classifier result, either probabilistic result or non-probabilistic result since transformations such as sigmoid can easily convert scores into fixed range..

6.1 Small Labeled Data Set

We illustrate that *fpr* does has advantage over other criteria in that it provides more consistent estimation of the threshold on a set of human judged documents. This set contains 4000 labeled page 404 documents. Page 404 indicates that the requested webpage was not found on the server, while the server is available for the client to communicate with. We apply the t-test to measure whether or not the thresholds determined by different criteria are statistically close to the true value. The procedures are as follows.

First, we apply *fpr*, precision and F_1 measure as the selection criteria to get the respective thresholds on entire labeled data set. By setting $fpr = 0.02$ and precision=95%, we obtain the results shown in Table 2. Note that 95% precision has actual precision of 100%. This is because 1) we do not allow the final precision to be lower than 95%; 2) for this case, adding extra error point, i.e. one false positive, causes the precision to drop below 95% precision because number of true positives so far is small.

Table 2. Threshold on entire 4000 page 404 documents

Criteria	Threshold	F_1 Measure	Precision(95%)	Recall	Error	fpr
Best Fbeta	112	0.5	0.444	0.573	0.082	0.056
$fpr = 0.02$	149	0.453	0.593	0.366	0.064	0.02
95% precision	223	0.015	1.000	0.008	0.071	0.00

Then we sample the set 50 times with replacement for three runs. In the first run, we sample with replacement 50 times, and for each time we sample 300 data points. In the second run, we also sample 50 times, but with 500 data points per set. The last run consists of 50 sampled sets with 800 data points each. For each set in every run, we calculate the threshold for each criterion. We summarize the results in Table 3.

Table 3. Statistics of the thresholds on the sampled page 404 documents

Sample Set	Mean	STD	Precision	Recall	Error	F_1 Measure
set300bF1	124.2	15.89	0.564	0.566	0.067	0.554
set500bF1	126.7	16.44	0.536	0.530	0.068	0.523
set800bF1	123.4	15.83	0.513	0.531	0.073	0.511
set300fpr	147.9	17.14	0.609	0.386	0.061	0.466
set500fpr	146.2	13.15	0.593	0.385	0.061	0.464
set800fpr	148.8	8.78	0.597	0.361	0.063	0.448
set300pr	198.56	19.60	0.925	0.156	0.064	0.245
set500pr	203.46	16.48	0.924	0.114	0.064	0.188
set800pr	200.9	15.58	0.896	0.124	0.065	0.203

In Table 3, the row for set300bF1 corresponds to the thresholding results on the run consisting of the sample sets of 300 points each, with best F_1 measure as the criterion. Here we convert the classification scores from [0, 1] to integers from 0 to 255 since it can be stored in one byte. Similarly the row for set300fpr corresponds to the results with *fpr=0.02* as the criterion, and that for set300pr contains the results with precision 95% as the criterion. Same explanation can be applied to other rows.

From Table 3 we can see that thresholds obtained based on *fpr* generally have small standard deviation except for the case of 300 sample size where set300bF1 has smallest standard deviation. Note that both set*n*00bF1 and set*n*00pr have relatively large standard deviation even when the sample size increases, while for set*n*00fpr the standard deviation decreases dramatically as the sample size increase. Therefore we think the higher standard deviation of set300fpr compared to set300bF1 is due to inadequate sample size.

The observation from this experiment is that

- Both best F_1 measure and precision criteria fail the significance test. The most probable reason is that they are sensitive to the change of underlying distribution. Remember that in the editorial test set, the number of positives is small, therefore randomly sampling portion of the data will likely result in significant change in distribution.
- *fpr* is a more stable criterion when the class distribution in the population is highly skewed.

7 Conclusion

In this paper, we propose a novel cost effective approach of threshold selection method for binary webpage classification with highly imbalanced class distribution. Due to limited budget on human judgment, we apply stratified sampling on the webpages to construct a very small sample set. The human judged samples are expanded to reflect the true class distribution of the webpage population. For the problems of unbalanced distribution and sparsity of positives in the sample set, experimental results show that false positive rate based threshold selection criterion gives more stable threshold than widely used precision and F_1 measure based approaches. It also let user control the type I error for classification.

Acknowledgement. The work is partially supported by the Key Program of National Natural Science Foundation of China (Grant No.61232002), National High Technology Research and Development Program 863 (Grant No.2012AA011003), National Natural Science Foundation of China (Grant No.61103039) .

References

1. Bennett, P.N.: Assessing the calibration of naive bayes posterior estimates. Tech. rep., Computer science department, school of computer science, CMU (2000)
2. Cochran, W.G.: Sampling Techniques, 3rd edn. John Wiley & Sons Inc., New York (1977)
3. Cohen, W.W., Singer, Y.: Context-sensitive learning methods for text categorization. ACM Transactions on Information Systems 17(2), 141–173 (1996)
4. Egan, J.P.: Signal Detection Theory and Roc Analysis. Academic Press, New York (1975)
5. Fawcett, T.: Draft roc graphs: Notes and practical considerations for data mining researchers (2003)
6. Ghani, R., Slattery, S., Yang, Y.: Hypertext categorization using hyperlink patterns and meta data. In: Proceedings of ICML 2001, 18th International Conference on Machine Learning, pp. 178–185. Morgan Kaufmann Publishers (2001)
7. Gvert, N., Lalmas, M., Fuhr, N.: A probabillistic description-oriented approach for categorising web documents. In: Proceedings of CIKM 1999, 8th ACM International Conference on Information and Knowledge Management, pp. 475–482 (1999)
8. Iwayama, M., Tokunaga, T.: Cluster-based text categorization: a comparison of category serch strategies. In: Proceedings of SIGIR 1995, 18th ACM International Conference on Research and Development in Information Retrieval, pp. 273–281. ACM Press (1995)
9. Larkey, L.S.: Automatic essay grading using text categorization techniques. In: Proceedings of SIGIR 1998, 21st ACM International Conference on Research and Development in Information Retrieval, pp. 90–95 (1998)
10. Levy, P.S., Lemeshow, S.: Sampling of Populations: Methods and Applications, 3rd edn. John Wiley & Sons Inc., New York (1999)
11. Lewis, D.D.: An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of SIGIR 1992, 15th ACM International Conference on Research and Development in Information Retrieval, pp. 37–50. ACM Press (1992)
12. Lewis, D.D.: Evaluating and optimizing autonomous text classification systems. In: Proceedings of SIGIR 1995, 18th ACM International Conference on Research and Development in Information Retrieval, pp. 246–254. ACM Press (1995)
13. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in Large Margin Classifiers, pp. 61–74. MIT Press (1999)
14. Schapire, R.E., Singer, Y., Singhal, A.: Boosting and rocchio applied to text filtering. In: Proceedings of ACM SIGIR 1998, 21st ACM International Conference on Research and Development in Information Retrieval, pp. 215–223. ACM Press (1998)
15. Sebastiani, F., Ricerche, C.N.D.: Machine learning in automated text categorization. ACM Computing Surveys 34, 1–47 (2002)
16. Spackman, K.A.: Signal detection theory: Valuable tools for evaluating inductive learning. In: Proceedings of 6th International Workshop on Machine Learning, pp. 160–163. Morgan Kaufman (1989)
17. Swets, J.A., Dawes, R.M., Monahan, J.: Context-sensitive learning methods for text categorization, pp. 82–87. Scientific American (October 2000)
18. Thompson, S.K.: Sampling, 2nd edn. John Wiley & Sons Inc., New York (2002)

19. Tie-Yan Liu, Yiming Yang, H.W.H.J.Z.Z.C., Ma, W.Y.: Support vector machines classification with a very large-scale taxonomy. In: ACM SIGKDD Explorations Newsletter - Natural Language Processing and Text Mining, vol. 7, pp. 36–43. ACM Press (June 2005)
20. Yang, Y.: An evaluation of statistical approaches to text categorization. Journal of Information Retrieval 1, 67–88 (1999)
21. Yang, Y.: A study on thresholding strategies for text categorization. In: Proceedings of SIGIR 2001, 24th ACM International Conference on Research and Development in Information Retrieval, pp. 137–145. ACM Press (2001)
22. Yang, Y., Slattery, S.A.: A study of approaches to hypertext categorization. Journal of Intelligent Information Systems 18, 219–241 (2002)

Author Index