

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

New York University, NY, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Wenfei Fan Zhaohui Wu Jun Yang (Eds.)

Advances in Web-Age Information Management

6th International Conference, WAIM 2005
Hangzhou, China, October 11 – 13, 2005
Proceedings

Volume Editors

Wenfei Fan
University of Edinburgh
School of Informatics
Appleton Tower, Crichton Street
Edinburgh EH8 9 LE, Scotland, UK
E-mail: wenfei@inf.ed.ac.uk

Zhaohui Wu
Zhejiang University
College of Computer Science and Technology
Hangzhou, Zhejiang, China
E-mail: wzh@cs.zju.edu.cn

Jun Yang
Duke University
Department of Computer Science
Durham, North Carolina 27708-0129, USA
E-mail: junyang@cs.duke.edu

Library of Congress Control Number: 2005932897

CR Subject Classification (1998): H.2, H.3, H.4, I.2, H.5, C.2, J.1

ISSN 0302-9743
ISBN-10 3-540-29227-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-29227-2 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springeronline.com

© Springer-Verlag Berlin Heidelberg 2005
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11563952 06/3142 5 4 3 2 1 0

Foreword by Conference Chairs

WAIM 2005 is the latest edition of the International Conference on Web-Age Information Management. Built on the successes of the past five WAIM conferences, WAIM 2005 takes another significant step in making WAIM conference a high-caliber international conference in the area of Web information management and database management. This year, WAIM received 486 submissions from 20 countries and regions.

In response to the record number of paper submissions WAIM 2005 received, the Program Co-chairs, Wenfei Fan and Zhaohui Wu, and the members of the Program Committee worked extremely hard to review all the submitted papers and select the best ones for inclusion in the conference proceedings. Their hard work has produced an excellent technical program for WAIM 2005. The Program Committee, together with Industrial Track Chair Phil Bohannon, Exhibition and Demo Chair Ying Jianwei, Tutorial Chair X. Sean Wang, and Panel Chair Jeffrey Xu Yu, has put together a very high-quality conference program. Jun Yang, the Publication Chair of WAIM 2005, also worked diligently to produce the conference proceedings.

The success of the conference would not have been possible without the hard work of many people and their contributions are greatly appreciated. Qinming He and Xiaohong Jiang are our local arrangement co-chairs. Ling Chen is the financial and registration chair. Xiaofang Zhou, Floris Geerts and Haixun Wang are the publicity co-chairs. Our special thanks go to the two Webmasters of the conference website, Yu Zhang and Mengya Tang, for their tremendous effort in facilitating the entire paper submission/reviewing process and in keeping the website running smoothly. We would also like to thank the Database Society of China Computer Federation, the Natural Science Foundation of China, Y. C. Tang Disciplinary Development Fund of Zhejiang University, and Oracle China for sponsoring WAIM 2005.

We would like to join the Program Committee and the entire database community in paying our tribute to Hongjun Lu, who passed away in March this year. Among his many outstanding contributions to the database community, Hongjun was instrumental in founding the WAIM conference and ensuring its success. He was heavily involved in the early preparation of WAIM 2005. Even when he became very ill, he still paid close attention to the progress of WAIM 2005 preparation. We owe it to him to make WAIM a premier conference in the database community.

Finally, we would like to use this opportunity to thank X. Sean Wang, who succeeded Hongjun as the Steering Committee Liaison to WAIM 2005, for his able guidance.

October 2005

Changjie Tang, Weiyi Meng
Conference Co-chairs
WAIM 2005

Foreword by Program Chairs

This volume contains the proceedings of the Sixth International Conference on Web-Age Information Management (WAIM), held in Hangzhou, China on October 11–13, 2005. This year we are pleased to have a particularly strong program: the Program Committee selected 48 papers for presentation and inclusion in the proceedings, and 50 short papers for inclusion in the proceedings, from 486 submissions, a record number, from 20 countries and regions. Acceptance into the conference proceedings was extremely competitive. The contributed papers span the range of traditional database topics as well as issues of emerging interest in connection with Web databases and services. The Program Committee worked hard to select these papers through a detailed two-phase review process and extensive discussion via a conference management tool, and put together a diverse and exciting program.

In addition to the research track, we had 20 submissions to the industrial track, which focuses on the early dissemination of prototype and system development experience. The Industrial Committee, chaired by Philip Bohannon, selected 3 papers for presentation and inclusion in the proceedings, and 1 short paper for inclusion in the proceedings. The Industrial program is a key component of this year's conference.

Papers accompanying each of the invited talks, by Peter Buneman (keynote), Hai Zhuge and Hai Jin, are included in this volume.

The success of WAIM 2005 is a concrete step toward establishing the status of WAIM as one of the leading international forums for database researchers, practitioners, developers and users to explore cutting-edge ideas and theories, and to exchange techniques, tools and experiences.

The Program Committee and the Industrial Committee thank all those who submitted their best work to the conference. On their behalf, we thank their colleagues, listed separately, who helped in the review process, and the sponsoring organizations, the Database Society of the China Computer Federation, the Natural Science Foundation of China, Y. C. Tang Disciplinary Development and Fund of Zhejiang University, and Oracle China. We also thank the local organizers and volunteers in Hangzhou for their effort and time to help make the conference a success.

As program co-chairs we would like to thank the members of the Program Committee for the hard work they put in both in writing the detailed reviews and for their participation in the active discussions. We also thank them for bearing with the primitive conference management tool. In particular, we would like to extend our special thanks to Gao Cong, Irimi Fundulaki, Floris Geerts, Anastasios Kementsietsidis, Weiyi Meng, Changjie Tang, X. Sean Wang and Stratis D. Viglas, who worked extremely hard to conduct the first phase of the review process; each of them reviewed at least 60 submissions, and their efforts

are essential to the quality of the program. We are grateful to Weiyi Meng, X. Sean Wang, Stratis D. Viglas, and Jeffrey Xu Yu for their valuable help in leading and monitoring the discussions on our behalf. We also thank Jun Yang for his work in preparing the proceedings. Finally, we are deeply indebted to Yu Zhang and Mengya Tang, the Web Masters, who took on tremendous pain and extra work, and ably modified, extended and maintained the conference management tool; for three long months they worked until late night every day, seven days a week; the necessary extension of the tool also received help from Kun Jing, Chengchao Xie, Shiqi Peng, Heng Wang, Cheng Jin and Ruizhi Ye; without their hard and effective work the online discussions would not have been possible, among other things.

October 2005

Wenfei Fan, Zhaohui Wu
Program Committee Co-chairs
WAIM 2005

Dedication: Hongjun Lu (1945–2005)



On behalf of the Program Committee, it is with sincere gratitude and great sorrow that we would like to dedicate WAIM 2005 proceedings to Hongjun Lu, who left us on March 3, 2005. Hongjun was not only an excellent researcher and highly productive scholar of the database community, but also a wonderful colleague and dear friend. For many years, he has been the ambassador for database research to China, and tremendously fostered the growth of this community. Among other things, Hongjun initiated, organized and monitored WAIM conferences in the past 5 years, and he was personally involved in the early stage of WAIM 2005 organization despite his health condition. Together with all the authors and all the members of the Program Committee, we would like to dedicate the proceedings to Hongjun as a tribute!

October 2005

Wenfei Fan, Zhaohui Wu
Program Committee Co-chairs
WAIM 2005

Organization

WAIM 2005 is organized by Zhejiang University (Hangzhou, China) in cooperation with the Database Society of China Computer Federation. It was also sponsored by Natural Science Foundation of China and Oracle China.

Organizing Committee

General Co-chairs:	Weiye Meng, State Univ. of New York, Binghamton (USA) Changjie Tang, Sichuan Univ. (China)
Program Co-chairs:	Wenfei Fan, Univ. of Edinburgh (UK) and Bell Labs (USA) Zhaohui Wu, Zhejiang Univ. (China)
Industrial-Track Chair:	Philip Bohannon, Bell Labs (USA)
Tutorial Chair:	X. Sean Wang, Univ. of Vermont (USA)
Panel Chair:	Jeffrey Xu Yu, Chinese Univ. of Hong Kong (China)
Demonstrations Chair:	Jianwei Yin, Zhejiang Univ. (China)
Steering Committee Liaison:	Hongjun Lu, Hong Kong Univ. of Science & Tech. (China) X. Sean Wang, Univ. of Vermont (USA)
Local Organization Co-chairs:	Qinming He, Zhejiang Univ. (China) Xiaohong Jiang, Zhejiang Univ. (China)
Publication Chair:	Jun Yang, Duke Univ. (USA)
Financial & Registration Chair:	Ling Chen, Zhejiang Univ. (China)
Publicity Co-chairs:	Xiaofang Zhou, Univ. of Queensland (Australia) Floris Geerts, Univ. of Edinburgh (UK) Wang Haixun, IBM (USA)

Program Committee

Steven Blott	Dublin City Univ. (Ireland)
Philip Bohannon	Bell Labs (USA)
Stephane Bressan	National Univ. of Singapore (Singapore)
Chee Yong Chan	National Univ. of Singapore (Singapore)
Mei Che	CASC (China)
Qingzhang Chen	Zhejiang Univ. of Tech. (China)
William Cheung	Hong Kong Baptist Univ. (China)

Richard Connor	Univ. of Strathclyde (UK)
Guo-zhong Dai	Inst. of Software, CAS (China)
Alin Deutsch	UC San Diego (USA)
Guozhu Dong	Wright State Univ. (USA)
Jin-xiang Dong	Zhejiang Univ. (China)
Zhumei Dou	CETC (China)
Xiaoyong Du	Renmin Univ. of China (China)
Jianhua Feng	Tsinghua Univ. (China)
Irini Fundulaki	Bell Labs (USA)
Sumit Ganguly	IIT (India)
Floris Geerts	Univ. of Edinburgh (UK)
Michael Gertz	UC Davis (USA)
Cong Gao	Univ. of Edinburgh (UK)
Yanbo Han	Inst. of Computing Tech., CAS (China)
Yanxiang He	Wuhan Univ. (China)
Joshua Huang	Hong Kong Univ. (China)
Yoshiharu Ishikawa	Univ. of Tsukuba (Japan)
Yan Jia	National Univ. of Defence Tech. (China)
Hai Jin	Huazhong Univ. of Science and Tech. (China)
Raghav Kaushik	Micorsoft (USA)
Anastasios Kementsietsidis	Univ. of Edinburgh (UK)
Hiroyuki Kitagawa	Univ. of Tsukuba (Japan)
Masaru Kitsuregawa	Univ. of Tokyo (Japan)
Yannis Kotidis	AT&T Labs-Research (USA)
Laks VS Lakshmanan	Univ. of British Columbia (Canada)
Dongwon Lee	Pennsylvania State Univ. (USA)
Chen Li	UC Irvine (USA)
Chengkai Li	UIUC (USA)
Dapeng Li	Shanghai Futures Exchange (China)
Minglu Li	Shanghai Jiaotong Univ. (China)
Jianzhong Li	Harbin Inst. of Tech. (China)
Jinyan Li	Inst. for Infocomm Research (Singapore)
Zhanhuai Li	Northwestern Polytechnic Univ. (China)
Daniel Lieuwen	Bell Labs (USA)
Xuemin Lin	Univ. of New South Wales (Australia)
Weiyi Liu	Yunnan Univ. (China)
Dianfu Ma	Beihang Univ. (China)
Xiaofeng Meng	Renmin Univ. of China (China)
Gu Ning	Fudan Univ. (China)
Werner Nutt	Heriot-Watt Univ. (UK)
Beng Chin Ooi	National Univ. of Singapore (Singapore)
Zhiyong Peng	Wuhan Univ. (China)
Rajeev Rastogi	Bell Labs (India)
Prasan Roy	IBM (India)
Junyi Shen	Xi'an Jiaotong Univ. (China)

Jianwen Su	UC Santa Barbara (USA)
Changjie Tang	Sichuan Univ. (China)
Stratis D. Viglas	Univ. of Edinburgh (UK)
Guoren Wang	Northeastern Univ. (China)
Haiyang Wang	Shandong Univ. (China)
Qingxian Wang	Zhengzhou Univ. of Info. Engineering (China)
X. Sean Wang	Univ. of Vermont (USA)
Shan Wang	Renmin Univ. of China (China)
Wei Wang	Univ. of North Carolina at Chapel Hill (USA)
Limsoon Wong	Inst. for Infocomm Research (Singapore)
Peter T. Wood	Univ. of London (UK)
Ming Xiong	Bell Labs (USA)
Dongqing Yang	Peking Univ. (China)
Ge Yu	Northeastern Univ. (China)
Jeffrey Xu Yu	Chinese Univ. of Hong Kong (China)
Lihua Yue	Univ. of Science and Tech. (China)
Aoying Zhou	Fudan Univ. (China)
Lizhu Zhou	Tsinghua Univ. (China)
Xiaofang Zhou	Univ. of Queensland (Australia)
Yueting Zhuang	Zhejiang Univ. (China)
Jin Zhi	Inst. of Mathematics, CAS (China)
Hai Zhuge	Inst. of Computing Tech., CAS (China)

Referees

Ali Al-Lawati	Helen Huang	Pabitra Mitra
Toshiyuki Amagasa	Huan Huo	Anirban Mondal
Yanrong Cai	Noriko Imafuji	Seog-Chan Oh
Julie Chang	Markus Jakobsson	Yuwei Peng
Lijun Chen	Ou Jianbo	Weining Qian
Wei Chen	Iosif Lazaridis	Ganesh Ramesh
Juntao Cui	Cao Lei	Yi Ren
Ken Deng	Aiping Li	Carlos Rueda
Zhiming Ding	Haifei Li	Takeshi Sagara
Michael Flaster	Xin Li	Derong Shen
Cagdas Gerede	Xuhui Li	Derong Shen
Sipei Guo	Jing Liping	Jiali Shen
Yang Guo	Jiamao Liu	Zhongnan Shen
WeiHong Han	Junqiang Liu	Quan Z. Sheng
Ramaswamy Hariharan	Qing Liu	Norihide Shinagawa
Quinn Hart	Zheng Liu	Houtan Shirani-Mehr
Bijit Hore	Yi Luo	Michal Shmueli-Scheuer
David Horgan	Yuxin Mao	Shefali Singhal
He Hu	Weiyi Meng	Katsumi Takahashi

XIV Organization

Changjie Tang
Feilong Tang
Meng Teng
Daling Wang
Di Wang
Hongzhi Wang
Jianyong Wang
Liwei Wang
Xiaoling Wang

Yitong Wang
Tok Wee Hyong
Li Xin
Chen Yan
Jiangming Yang
Nan Yang
Wei Ye
Bill Yu Zhang
Kun Yue

Cheng Zeng
Ende Zhang
Jie Zhang
Shichao Zhang
Xiaofeng Zhang
Zhongfei Zhang
Bin Zhou

Table of Contents

Keynote and Invited Talks

What the Web Has Done for Scientific Data – and What It Hasn't <i>Peter Buneman</i>	1
Integrity Theory for Resource Space Model and Its Application <i>Hai Zhuge, Yunpeng Xing</i>	8
Challenges of Grid Computing <i>Hai Jin</i>	25

Research Session 1: XML

BBTC: A New Update-Supporting Coding Scheme for XML Documents <i>Jianhua Feng, Guoliang Li, Lizhu Zhou, Na Ta, Qian Qian, Yuguo Liao</i>	32
Using XML Structure to Reduce Candidate Nodes Participated in Query Processing <i>Zhenying He, Jianzhong Li, Chaokun Wang, Pengcheng Ge, Haikun Chen</i>	45
An Effective and Efficient Approach for Keyword-Based XML Retrieval <i>Xiaoguang Li, Jian Gong, Daling Wang, Ge Yu</i>	56
Subgraph Join: Efficient Processing Subgraph Queries on Graph-Structured XML Document <i>Hongzhi Wang, Wei Wang, Xuemin Lin, Jianzhong Li</i>	68

Research Session 2: Performance and Query Evaluation

A Soft Real-Time Web News Classification System with Double Control Loops <i>Huayong Wang, Yu Chen, Yiqi Dai</i>	81
Searching the World Wide Web for Local Services and Facilities: A Review on the Patterns of Location-Based Queries <i>Saeid Asadi, Chung-Yi Chang, Xiaofang Zhou, Joachim Diederich</i> ..	91

Self-adaptive Statistics Management for Efficient Query Processing
Xiaojing Li, Gang Chen, Jinxiang Dong, Yuan Wang 102

Design and Implementation of the Modified R-Tree Structure with
 Non-blocking Querying
*Myungkeun Kim, Sanghun Eo, Seokkyu Jang, Jaedong Lee,
 Haeyoung Bae* 114

Research Session 3: Data Mining (I)

Importance-Based Web Page Classification Using Cost-Sensitive SVM
Wei Liu, Gui-rong Xue, Yong Yu, Hua-jun Zeng 127

An Efficient Approach for Interactive Mining of Frequent Itemsets
Zhi-Hong Deng, Xin Li, Shi-Wei Tang 138

Similarity Search with Implicit Object Features
Yi Luo, Zheng Liu, Xuemin Lin, Wei Wang, Jeffrey Xu Yu 150

An Improved FloatBoost Algorithm for Naïve Bayes Text Classification
*Xiaoming Liu, Jianwei Yin, Jinxiang Dong,
 Memon Abdul Ghafoor* 162

Research Session 4: Semantic Web and Web Ontology

An Approach to RDF(S) Query, Manipulation and Inference on
 Databases
Jing Lu, Yong Yu, Kewei Tu, Chenxi Lin, Lei Zhang 172

Clustering OWL Documents Based on Semantic Analysis
Mingxia Gao, Chunlian Liu, Furong Chen 184

An Ontology Based Approach to Construct Behaviors in Web
 Information Systems
*Lu-an Tang, Hongyan Li, Zhiyong Pan, Dongqing Yang, Meimei Li,
 Shiwei Tang, Ying Ying* 194

A Semi-automatic Ontology Acquisition Method for the Semantic Web
Man Li, Xiaoyong Du, Shan Wang 209

Research Session 5: Data Management (I)

Watermarking Abstract Tree-Structured Data
Gang Chen, Ke Chen, Tianlei Hu, Jinxiang Dong 221

Location-Based Caching Scheme for Mobile Clients <i>KwangJin Park, MoonBae Song, Chong-Sun Hwang</i>	233
Extended Derivation Cube Based View Materialization Selection in Distributed Data Warehouse <i>Wei Ye, Ning Gu, Genxing Yang, Zhenyu Liu</i>	245
A Framework of HTML Content Selection and Adaptive Delivery <i>Chen Ding, Shutao Zhang, Chi-Hung Chi</i>	257

Research Session 6: Information Systems

An Improved Multi-stage (t, n)-Threshold Secret Sharing Scheme <i>Hui-Xian Li, Chun-Tian Cheng, Liao-Jun Pang</i>	267
Information Management in E-Learning System <i>Liu Jing, Zheng Li, Yang Fang</i>	275
A Stratification-Based Approach to Accurate and Fast Image Annotation <i>Jianye Ye, Xiangdong Zhou, Jian Pei, Lian Chen, Liang Zhang</i>	284

Research Session 7: Web Services and Workflow (I)

Web Services Composition Based on Ontology and Workflow <i>Huaizhou Yang, Zengzhi Li, Jing Chen, Hong Xia</i>	297
Web Service Composition Using Markov Decision Processes <i>Aiqiang Gao, Dongqing Yang, Shiwei Tang, Ming Zhang</i>	308
FECT: A Modelling Framework for Automatically Composing Web Services <i>Lishan Hou, Zhi Jin</i>	320
Intrusion Detection of DoS/DDoS and Probing Attacks for Web Services <i>Jun Zheng, Ming-zeng Hu</i>	333

Research Session 8: Data Management (II)

cGridex: Efficient Processing of Continuous Range Queries over Moving Objects <i>Xiaoyuan Wang, Qing Zhang, Weiwei Sun, Wei Wang, Baile Shi</i> . . .	345
---	-----

An Efficient Context Modeling and Reasoning System in Pervasive Environment: Using Absolute and Relative Context Filtering Technology
Xin Lin, Shanping Li, Jian Xu, Wei Shi, Qing Gao 357

Influences of Functional Dependencies on Bucket-Based Rewriting Algorithms
Qingyuan Bai, Jun Hong, Hui Wang, Michael F. McTear 368

GARWM: Towards a Generalized and Adaptive Watermark Scheme for Relational Data
Tian-Lei Hu, Gang Chen, Ke Chen, Jin-Xiang Dong 380

Research Session 9: Data Grid and Database Languages

Refined Rules Termination Analysis Through Activation Path
Zhongmin Xiong, Zhongxiao Hao 392

Stream Operators for Querying Data Streams
Lisha Ma, Stratis D. Viglas, Meng Li, Qian Li 404

Join Algorithm Using Multiple Replicas in Data Grid
Donghua Yang, Jianzhong Li, Qaisar Rasool 416

Research Session 10: Agent and Mobile Data

A Heuristic and Distributed QoS Route Discovery Method for Mobile Ad Hoc Networks
Peng Fu, Deyun Zhang 428

Planning Enhanced Grid Workflow Management System Based on Agent
Lei Cao, Minglu Li, Jian Cao, Ying Li 440

Reasoning Based on the Distributed β -PSML
Yila Su, Jiming Liu, Ning Zhong, Chunnian Liu 452

Research Session 11: Data Mining (II)

An Auto-stopped Hierarchical Clustering Algorithm Integrating Outlier Detection Algorithm
Tian-yang Lv, Tai-xue Su, Zheng-xuan Wang, Wan-li Zuo 464

Research Paper Recommender Systems: A Subspace Clustering Approach <i>Nitin Agarwal, Ehtesham Haque, Huan Liu, Lance Parsons</i>	475
Concept Updating with Support Vector Machines <i>Yangguang Liu, Qinming He</i>	492
On the Performance of Feature Weighting K -Means for Text Subspace Clustering <i>Liping Jing, Michael K. Ng, Jun Xu, Joshua Zhexue Huang</i>	502
Research Session 12: Web Services and Workflow (II)	
State Transfer Graph: An Efficient Tool for Webview Maintenance <i>Yan Zhang, Xiangdong Qin</i>	513
An EJB-Based Very Large Scale Workflow System and Its Performance Measurement <i>Kwang-Hoon Kim, Hyong-Jin Ahn</i>	526
Deploying π -Calculus Technology in Inter-organizational Process <i>Memon Abdul Ghafoor, Jianwei Yin, Jinxiang Dong, Maree Mujeeb-u-Rehman</i>	538
Modulation for Scalable Multimedia Content Delivery <i>Henry Novianus Palit, Chi-Hung Chi</i>	556
Research Session 13: Database Application and Transaction Management	
A Deadline-Sensitive Approach for Real-Time Processing of Sliding Windows <i>Shanshan Wu, Ge Yu, Yaxin Yu, Zhengyu Ou, Xinhua Yang, Yu Gu</i>	566
MPX: A Multiversion Concurrency Control Protocol for XML Documents <i>Yuan Wang, Gang Chen, Jin-xiang Dong</i>	578
An Index Structure for Parallel Processing of Multidimensional Data <i>Kyoungho Bok, DongMin Seo, SeokIl Song, MyoungHo Kim, JaeSoo Yoo</i>	589

Industrial Session

Construction of Security Architecture of Web Services Based EAI
Di Wu, Yabo Dong, Jian Lin, Miaoliang Zhu 601

Using Web Services and Scientific Workflow for Species Distribution
 Prediction Modeling
Jianting Zhang, Deana D. Pennington, William K. Michener 610

Integrative Security Management for Web-Based Enterprise
 Applications
*Chen Zhao, Yang Chen, Dawei Xu, NuerMaimaiti Heilili,
 Zuoquan Lin* 618

Short Papers and Demonstration

An Ontology-Based Semantic Integration for Digital Museums
Hong Bao, Hongzhe Liu, Jiehua Yu, Hongwei Xu 626

A Unified Subspace Outlier Ensemble Framework for Outlier Detection
Zengyou He, Shengchun Deng, Xiaofei Xu 632

A Convertible Limited Verifier Signature Scheme
Jianhong Zhang, Hu'an Li, Jilin Wang 638

Short Signature Scheme Based on Discrete Logarithms
Zuhua Shao 645

Simulating a Finite State Mobile Agent System
Yong Liu, Congfu Xu, Yanyu Chen, Yunhe Pan 651

Algorithm for Analyzing N-Dimensional Hilbert Curve
Chenyang Li, Yucai Feng 657

DMT: A Flexible and Versatile Selectivity Estimation Approach for
 Graph Query
Jianhua Feng, Qian Qian, Yuguo Liao, Guoliang Li, Na Ta 663

A New Public Key Certificate Revocation Scheme Based on One-Way
 Hash Chain
JingFeng Li, YueFei Zhu, Heng Pan, DaWei Wei 670

Interactive Chinese Search Results Clustering for Personalization
Wei Liu, Gui-Rong Xue, Shen Huang, Yong Yu 676

Efficient Delay Aware Peer-to-Peer Overlay Network <i>Da-lu Zhang, Chen Lin</i>	682
PIES: A Web Information Extraction System Using Ontology and Tag Patterns <i>Byung-Kwon Park, Hyoil Han, Il-Yeol Song</i>	688
An Algebraic Framework for Schema Matching <i>Zhi Zhang, Haoyang Che, Pengfei Shi, Yong Sun, Jun Gu</i>	694
A Clustering Algorithm Based Absorbing Nearest Neighbors <i>Jian-jun Hu, Chang-jie Tang, Jing Peng, Chuan Li, Chang-an Yuan, An-long Chen</i>	700
Parallel Mining of Top-K Frequent Itemsets in Very Large Text Database <i>Yongheng Wang, Yan Jia, Shuqiang Yang</i>	706
Removing Smoothing from Naive Bayes Text Classifier <i>Wang-bin Zhu, Ya-ping Lin, Mu Lin, Zhi-ping Chen</i>	713
Medical Image Clustering with Domain Knowledge Constraint <i>Haiwei Pan, Jianzhong Li, Wei Zhang</i>	719
Tick Scheduling: A Deadline Based Optimal Task Scheduling Approach for Real-Time Data Stream Systems <i>Zhengyu Ou, Ge Yu, Yazin Yu, Shanshan Wu, Xiaochun Yang, Qingxu Deng</i>	725
A Novel Ranking Strategy in Hybrid Peer-to-Peer Networks <i>Qian Zhang, Zheng Liu, Xia Zhang, Xuezhi Wen, Yu Sun</i>	731
An Algorithmic Approach to High-Level Personalisation of Web Information Systems <i>Klaus-Dieter Schewe, Bernhard Thalheim</i>	737
sPAC (Web Services Performance Analysis Center): A Performance-Aware Web Service Composition Tool <i>Hyung Gi Song, Kangsun Lee</i>	743
Design and Implementation of a Service Bundle Manager for Home Network Middleware <i>Minwoo Son, Jonghwa Choi, Namhoon Kim, Dongkyoo Shin, Donggil Shin</i>	749

A State-Transfer-Based Dynamic Policy Approach for Constraints in RBAC <i>Cheng Zang, Zhongdong Huang, Gang Chen, Jinxiang Dong</i>	755
A New Cache Model and Replacement Algorithm for Network Attached Optical Jukebox <i>Xuan Liu, Tijun Lu, Huibo Jia</i>	761
Multiple Watermarking Using Hadamard Transform <i>Haifeng Li, Shuxun Wang, Weiwei Song, Quan Wen</i>	767
An Immunity-Based Intrusion Detection Solution for Database Systems <i>Ke Chen, Gang Chen, Jinxiang Dong</i>	773
Filtering Duplicate Items over Distributed Data Streams <i>Tian Xia, Cheqing Jin, Xiaofang Zhou, Aoying Zhou</i>	779
An Optimized K-Means Algorithm of Reducing Cluster Intra-dissimilarity for Document Clustering <i>Daling Wang, Ge Yu, Yubin Bao, Meng Zhang</i>	785
Hierarchical Metadata Driven View Selection for Spatial Query Evaluations <i>Songmei Yu</i>	791
Priority Processing in the Web Service-Workflow Architecture <i>Hwa-Young Jeong</i>	798
Policy-Based Workflow Management System <i>Shi Chen, Song Ouyang, G.K. Hassana</i>	804
A Mobile-Aware System for Website Personalization <i>S. Greco, A. Scicchitano, A. Tagarelli, E. Zumpano</i>	810
An Algorithm for Best Selection in Semantic Composition of Web Service* <i>Juntao Cui, Yiwen Zhu, Ning Gu, Yuwei Zong, Zhigang Ding, Shaohua Zhang, Quan Zhang</i>	816
Using Quantitative Association Rules in Collaborative Filtering <i>Xiaohua Sun, Fansheng Kong, Hong Chen</i>	822
Extracting, Presenting and Browsing of Web Social Information <i>Yi Wang, Li-zhu Zhou</i>	828

An Ontology-Based Host Resources Monitoring Approach in Grid Environment <i>Yijiao Yu, Hai Jin</i>	834
Domain-Specific Website Recognition Using Hybrid Vector Space Model <i>Baoli Dong, Guoning Qi, Xinjian Gu</i>	840
Understanding User Operations on Web Page in WISE <i>Hongyan Li, Ming Xue, Jianjun Wang, Shiwei Tang, Dongqing Yang</i>	846
Two-Phase Exclusion Based Broadcast Adaptation in Wireless Networks <i>Keke Cai, Huaizhong Lin, Chun Chen</i>	852
Web Service Collaboration Analysis via Automata <i>Yuliang Shi, Liang Zhang, Fangfang Liu, Lili Lin, Baile Shi</i>	858
A Novel Resource Description Based Approach for Clustering Peers <i>Xing Zhu, Dingyi Han, Yong Yu, Weibin Zhu</i>	864
The Segmentation of News Video into Story Units <i>Huayong Liu, Hui Zhang</i>	870
Process Controlling and Monitoring Scheme for Distributed Systems with Fault-Tolerance by Using Web Services <i>YunHee Kang, KyungWoo Kang</i>	876
CoopStreaming: A Novel Peer-to-Peer System for Fast Live Media Streaming <i>Jianwei Yin, Weipeng Yao, Lingxiao Ma, Jinxiang Dong</i>	882
Ontology-Based HTML to XML Conversion <i>Shijun Li, Weijie Ou, Junqing Yu</i>	888
Image Matrix Fisher Discriminant Analysis (IMFDA)- 2D Matrix Based Face Image Retrieval Algorithm <i>C.Y. Zhang, H.X. Chen, M.S. Chen, Z.H. Sun</i>	894
Approximate Common Structures in XML Schema Matching <i>Shengrui Wang, Jianguo Lu, Ju Wang</i>	900
Bi-directional Ontology Versioning BOV <i>Siyang Zhao, Brendan Tierney</i>	906
Quantitative Modeling for Web Objects' Cacheability <i>Chen Ding, Chi-Hung Chi, Lin Liu, LuWei Zhang, H.G. Wang</i>	913

An Efficient Scheme of Merging Multiple Public Key Infrastructures in ERP <i>Heng Pan, YueFei Zhu, ZhengYun Pan, XianLing Lu</i>	919
Forms-XML: Generating Form-Based User Interfaces for XML Vocabularies <i>Y.S. Kuo, N.C. Shih, Lendle Tseng, Hsun-Cheng Hu</i>	925
Author Index	927

What the Web Has Done for Scientific Data – and What It Hasn't

Peter Buneman

School of Informatics, University of Edinburgh and Digital Curation Centre

Abstract. The web, together with database technology, has radically changed the way scientific research is conducted. Scientists now have access to an unprecedented quantity and range of data, and the speed and ease of communication of all forms of scientific data has increased hugely. This change has come at a price. Web and database technology no longer support some of the desirable properties of paper publication, and it has introduced new problems in maintaining the scientific record. This brief paper is an examination of some of these issues.

1 Introduction

Try to imagine the unthinkable: you have lost your internet connection. So you go to the reference shelves of your local library for some information relevant to your work. Perhaps you are interested in demography and want the GDP and population of some country. The chances are that you will find a rather sorry and little-used collection of reference books, most of them relics of the time before the web – only a few years ago – when libraries were the main vehicle for the dissemination of scientific and scholarly information. The reference books have been replaced by databases to which, if they are not open-access, the library has subscribed on your behalf. You would, of course, be much better off using the web. In fact, for scientific data the web has had huge benefits

- it has provided access to much larger and richer data collections;
- the information is much more timely – we do not have to wait for a new edition to be printed to get an up-to-date GDP;
- access to the information is much faster and simpler;
- the information is better presented; and
- as a result, new information sources have been created which both classify scientific data in useful ways and form a vehicle for the communication of scientific opinion.

The impact of the web on the way scientific research is conducted has been enormous. Michael Lesk [Les] has argued that it has actually changed the scientific method from “hypothesize, design and run experiment, analyze results” to “hypothesize, look up answer in data base”. Almost all of modern science is now dependent on databases. Biology has led the way in the use of organised data collections to disseminate knowledge (I shall refer to these as databases) but

nearly all branches of scientific research are now dependent on web-accessible data resources. Databases are vehicles for publishing data (in fact the databases themselves can be considered publications), and it is often a condition of scientific funding that an appropriate database be generated and made accessible to the scientific community.

All this represents spectacular progress. We should not be upset that the library is no longer the primary vehicle for the dissemination of scholarship. But is it possible that in the rush to place our data “on the web” we are losing some important functions that libraries – whether or not by design – traditionally provided? Consider again your journey to the library.

First, if the reference work you were looking for is not in the library, the chances are that some other library has a copy of it. By having copies of the same work distributed among many libraries, there is some guarantee that the information is being preserved. Copying has always been the best guarantee of preservation. Now that your data is kept at some centralised database, it is not at all clear that it is in a form appropriate for long-term presentation. Also, the responsibility for keeping the information is now assumed by the organisation that maintains the database rather than being distributed among a number of libraries.

Second, maybe what you were looking for is in a reference book that is updated from time to time. If the library decided not to buy the new edition, at least you could revert to an old edition. Now, if the library drops its subscription to the on-line data, what do you have? This underlines the fact that the economic and intellectual property issues with databases on the web are very different from those that apply to traditional paper-based dissemination of knowledge. However the law that applies to digital data collections is effectively based on the traditional copyright laws.

Third, once you had found the information you were looking for, there was a serviceable method of citing it according to one of a few accepted methods. You could, if needed, localise the information by giving a page number, and the citation could be checked by other people who had access to the cited document. Now it is not at all clear how you cite something you find in a database; and you have no guarantee that it can be checked. Maybe the web site has disappeared, or maybe the database has been updated.

Fourth, the database keeps up-to-date information, but you might want some old information – perhaps the GDP from some past year. The old publications in the library may have this information, but the database does not.

These differences indicate that there are a number of problems with the dissemination of scientific data on the web. Having fast access to up-to-date research material may come at the price of data quality. Arguably, the web is losing the scientific record as fast as it is creating it; and users of web data have little faith in it until they can verify its provenance and authorship.

The rest of this paper is an attempt to show that, in trying to remedy these drawbacks of web data, we are led to some new and challenging problems that involve databases, the web and other areas of computer science.

1.1 Scientific Data

The use of database technology – in a broad sense of the term – to support scientific data is increasing rapidly. However, scientific data puts different demands on database technology. For example, transaction rates are typically lower in the maintenance of most scientific databases. Scale [HT] is arguably important, and complexity is surely important. Not only is “schema” complexity high, but the kinds of interactions between query languages and scientific programming require relatively complex algorithms and place new demands on the already complex area of query optimisation [Gra04]. The latter paper deals well with some of the issues of scale. In this note I want to deal with a largely orthogonal set of issues that have arisen in discussion with scientists who are dealing with databases in which the primary issues, at least for the time being, do not concern scale, but involve the manipulation, transfer, publishing, and long-term sustainability of data. Biological data has been the prime mover in some of these issues, but other sciences are catching up.

2 Data Transformation and Integration

Data integration, of course, a relatively old topic in database research, which is crucial to curated databases. While low-level tools such as query languages that can talk to a variety of data formats and databases are now well-developed; declarative techniques for integration and transformation based on schema annotation and manipulation have been slow to come to market; and where progress has been made it is with relatively simple data models [HHY⁺01, PB03]. A survey of the *status quo* is well beyond the scope of this paper, but it is worth remarking that while the emergence of XML as universal data exchange format may help in the low-level aspects of data integration through the use of common tools such as XQuery [XQu], it is not at all clear whether XML, has helped in the higher-level, schema based approach to data integration. The complexities of constraint systems such as XML Schema [XML] appear to defy any attempt at schema-based integration. Moreover, it is not clear what serialisation formats – upon which XML Schema is based – have to do with the data models in which people naturally describe data.

A more limited goal for XML research, and one in which progress has been made, is that of *data publishing*. There is again a growing literature on this topic. The idea is that individual organisations will maintain their data using a variety of models and systems but will agree to common formats, probably XML, for the exchange of data. The problem now is to export data efficiently in XML and, possibly, to transform and import the XML into other databases. This not only requires efficient and robust tools for describing and effecting the transformation [FTS00, BCF⁺02] but also tools for efficiently recomputing differences when the source database is modified – a new form of the view maintenance problem.

3 Data Citation

A common complaint from people who annotate databases based on what is in the printed literature is that citations are not specific enough. For example, in the process of verifying an entry in some biological database, one needs to check that a given article mentions a specific chemical compound. Even if one is given a page number, it can be quite time consuming to pinpoint the reference. The point here is that the more we can localise citations the better. Now consider the issues involved with citing something in a database. There are two important issues.

- How does one cite the database itself and localise the information within the database?
- How does one cope with the fact that the database itself changes? Does a change necessarily invalidate the citation?

As we know, URLs and URIs fail to meet the needs of stable “coarse grained” identifiers, such as identifiers of a database or web site. This has led people interested in long-term preservation to consider a variety of techniques for maintaining digital object identifiers that persist over an extended period. But even when the domain of citation is in our control, for example we want to specify localised citations within a website or database, how do we specify these citations, and what whose responsibility is it to deal with changing data? For relational databases, a solution to the localisation problem is to use keys or system tuple identifiers. For hierarchical data such as XML, a solution is suggested in [BDF⁺02]. However these are partial solutions to the localisation problem. Standards for data citation need to be developed, and dealing with change is a major problem.

4 Annotation

This is a growing area of activity in scientific databases. Some biological databases describe themselves as “annotation databases”, and there are some systems [SED] which are designed to display an overlay of annotations on existing databases. Database management systems typically do not provide “room” for *ad hoc* or unanticipated annotation, and only recently has any attempt been made to understand what is required of database systems to provide this functionality [CTG04].

5 Provenance

Also known as “lineage” and “pedigree” [CW01, BKT00], this topic has a variety of meanings. Scientific programming often involves complex chains or workflows of computations. If only because one does not want to repeat some expensive analysis, it is important to keep a complete record of all the parameters of

a workflow and of its execution. This is sometimes known as “workflow” or “coarse-grained” provenance [SM03].

In curated databases, as we have already noted, data elements are repeatedly copied from one database to the next. As part of data quality know where a data element has come from, which databases it has passed through, and what actions or agents created and modified it. Even formulating a model in which it is possible to give precise definitions to these is non-trivial. Moreover, since much copying is done by programs external to the databases or by user actions, it is a major challenge to create a system in which provenance is properly recorded. It involves much more than database technology. For example, data is often copied by simple “copy-and-paste” operations from one database to another. To provide proper mechanisms for tracking this data movement will involve not only re-engineering the underlying database systems but also to the operating systems and interactive environments in which the changes are made.

6 Preservation

Keeping the past states of a database is an important part of the scientific record. Most of us have been “burned” by a reference to a web page or on-line publication that has disappeared or has been changed without any acknowledgment of the previous version. This is one area in which we have made some progress [BKTT04]. A system has been implemented which records every version of a database in a simple XML file. For a number of scientific databases on which this has been tested, the size of an archive file containing a year's worth of changes exceeds the size of one version of the database by about 15%. Yet any past version of the database may be retrieved from the archive by a simple linear scan of the archive.

The principle behind this system is that, rather than recording a sequence of versions, one records one database with the changes to each component or object recorded at a fine-grained level. This relies on each component of the database having a canonical location in the database, which is described by a simple path from the root of the database. It is common for scientific data to exhibit such an organisation, and this organisation may be of use in other aspects of curated data such as annotation, where some notion of “co-ordinate” or “key” is needed for the attachment of external annotation. In fact, it relies crucially on a system for fine-grain citation such as that advocated in section 3.

Archiving in this fashion does a little more than “preserve the bits”. For a relational database, it dumps the database into XML making it independent of a specific database management system and intelligible to someone who understands the structure of the data. The subject of digital preservation is more than preserving bits. It is about preserving the *interpretation* of a digital resource. For example, you have a document in the internal format of an early word processor. Should you be concerned about preserving the text, the formatted text, or the “look and feel” of the document as it was to the users of that word processor [CLB01, LMM01, OAI]. Databases may be in a better position

because there is arguably only one level of interpretation – the SQL interface for relational databases, or the syntactic representation of the data in XML. But this does not mean that there is nothing to worry about. How do you preserve the schema, and are there other issues of context that need to be maintained for the long-term understanding of the data?

7 Database Research Is Not Enough

Integration, annotation, provenance, citation, and archiving are just a few of the new topics that have emerged from the increasing use of curated databases. Some progress can be made by augmenting existing database technology. But fully to deal with provenance and integration, we need a closer integration of databases with programming languages and operating systems. These require better solutions to the impedance mismatch problem. Some progress was made with object-oriented databases, and more recently in programming languages [Com] and web programming [Wad] which understand typed data and in which file systems are replaced by database systems as a fundamental approach to solving the impedance mismatch problem. It is not clear whether the natural inertia in software development will ever allow such a radical change to take place. In addition, even if all the technical problems are solved, the social, legal and economic problems with web data are enormous.

References

- [BCF⁺02] Michael Benedikt, Chee Yong Chan, Wenfei Fan, Rajeev Rastogi, Shihui Zheng, and Aoying Zhou. DTD-directed publishing with attribute translation grammars. In *VLDB*, 2002.
- [BDF⁺02] Peter Buneman, Susan Davidson, Wenfei Fan, Carmem Hara, and Wang-Chiew Tan. Keys for XML. *Computer Networks*, 39(5):473–487, August 2002.
- [BKT00] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. Data Provenance: Some Basic Issues. In Sanjiv Kapoor and Sanjiva Prasad, editors, *Proceedings of FST TCS 2000: Foundations of Software Technology and Theoretical Computer Science*, pages 87 – 93. Springer, LNCS 1974, Dec 2000.
- [BKTT04] Peter Buneman, Sanjeev Khanna, Keishi Tajima, and Wang-Chiew Tan. Archiving scientific data. *ACM Transactions on Database Systems*, 27(1):2–42, 2004.
- [CLB01] James Cheney, Carl Lagoze, and Peter Botticelli. Toward a theory of information preservation. In *5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2001, Darmstadt, 2001*. Cw. <http://research.microsoft.com/Comega/>.
- [CTG04] L. Chiticariu, W-C. Tan, and G.Vijayvardiya. An annotation management system for relational databases. In *VLDB*, 2004.
- [CW01] Yingwei Cui and Jennifer Widom. Lineage tracing for general data warehouse transformations. In *Proc. of 27th International Conference on Very Large Data Bases (VLDB'01, Rome, Italy, September 2001*.

- [FTS00] Mary F. Fernandez, Wang Chiew Tan, and Dan Suci. SilkRoute: trading between relations and XML. *Computer Networks*, 33(1-6):723–745, 2000.
- [Gra04] Jim Gray. Distributed computing economics. In A. Herbert and K. Sparck Jones, editors, *Computer Systems Theory, Technology, and Applications, A Tribute to Roger Needham*, pages 93–101. Springer, 2004.
- [HHY⁺01] R.J. Millerand M.A. Hernández, L.M. Haas, L. Yan, C.T.H. Ho, R. Fagin, and L. Popa. The clio project: Managing heterogeneity. *SIGMOD Record*, 30(1), March 2001.
- [HT] Tony Hey and Anne Trefethen. The data deluge: An e-science perspective. http://www.rcuk.ac.uk/escience/documents/report_datadeluge.pdf referenced 20 July 2005.
- [Les] Michael Lesk. archiv.twoday.net/stories/337419/ referenced 22 July 2005.
- [LMM01] B. Ludäscher, R. Marciano, and R. Moore. Towards self-validating knowledge-based archives. In *11th Workshop on Research Issues in Data Engineering (RIDE)*, Heidelberg, Germany. IEEE Computer Society, April 2001.
- [OAI] Reference model for an open archival information system (oais). CCSDS 650.-B-1. Blue Book. Issue 1. Washington D.C. January 2002 <http://www.ccsds.org/documents/pdf/CCSDS-650.0-B-1.pdf>.
- [PB03] R.A. Pottinger. and P.A. Bernstein. Merging models based on given correspondences. In *VLDB*, 2003.
- [SED] Lincoln D. Stein, Sean Eddy, and Robin Dowell. Distributed Sequence Annotation System (DAS). <http://www.biodas.org/documents/spec.html>.
- [SM03] Martin Szomszor and Luc Moreau. Recording and reasoning over data provenance in web and grid services. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pages 63–620. Springer, LNCS 2888, 2003.
- [Wad] P. Wadler. The links project. <http://homepages.inf.ed.ac.uk/wadler/papers/links/links-blurb.pdf>
- [XML] XML Schema Part 1: Structures Second Edition. <http://www.w3.org/TR/xmlschema-1/>.
- [XQu] XQuery 1.0: An XML Query Language. <http://www.w3.org/TR/xquery/>.

Integrity Theory for Resource Space Model and Its Application¹

Hai Zhuge and Yunpeng Xing

China Knowledge Grid Research Group, Key Lab of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences, 100080, Beijing, China
Zhuge@ict.ac.cn, Ypxing@kg.ict.ac.cn
<http://kg.ict.ac.cn/>

Abstract. The Resource Space Model (RSM) is a semantic data model based on orthogonal classification semantics for effectively managing various resources in interconnection environment. In parallel with the integrity theories of relational and XML-based data models, this keynote presents the integrity theory for the RSM, including the entity integrity constraints based on the key system of the RSM, the membership integrity constraints, the referential integrity constraints, and the user-defined integrity constraints relevant to applications. This theory guarantees the RSM to correctly and efficiently specify and manage resources. Its implementation approach and application in culture exhibition are demonstrated.

1 Introduction

Integrity theory plays an important role in the relational data model, which has obtained a great success in both theory and systems [6, 7, 8, 10, 12, 15]. Database applications request that changes made to data by authorized users do not result in a loss of data consistency. Effective integrity constraints are means to fight the damage to data and to the consistency between data and its management mechanism. However, previous data models are limited in their abilities in effectively managing heterogeneous, distributed, and ocean resources in an open and dynamic Internet environment [17].

On the other hand, the success of World Wide Web guides people to attack the representation issue of resources. XML facilitates representation and exchange of structured information on the Internet. The Semantic Web steps further this way by using markup languages like RDF and establishing ontology mechanisms [2].

Much research on XML-based information organization and management has been done, such as Document Type Definitions, XML Schema and Unified Constraint Model for XML [3, 11, 16]. The integrity constraints for semantic specifications of XML data have drawn attention [9, 13]. However, the Internet still lacks ideal semantic data model to effectively manage distributed and expanding resources.

The Resource Space Model RSM is a semantic data model for uniformly, normally and effectively specifying and managing resources in interconnection environment.

¹ This work is supported by National Basic Research and Development Program (Semantic Grid Project, No. 2003CB317001) and the National Science Foundation of China.

Its theoretical basis is a set of normal forms on orthogonal classification semantics and the principles on relevant resource operations [17, 18, 19, 20].

Integrity is essential for the RSM to ensure its classification semantics, to maintain the consistency on resource spaces, to optimize queries, and to correctly and efficiently manage resources.

2 Resource Space System

The architecture of the resource space system includes five layers as depicted in Fig. 1: the application layer, the resource space system layer, the resource representation layer representing semantics of resources, the entity resource layer, and the network layer. Here focuses on the resource space system layer, which includes the resource space layer organizing resources according to semantic normal forms [17] and the operation mechanism layer implementing the operations on resources and resource spaces.

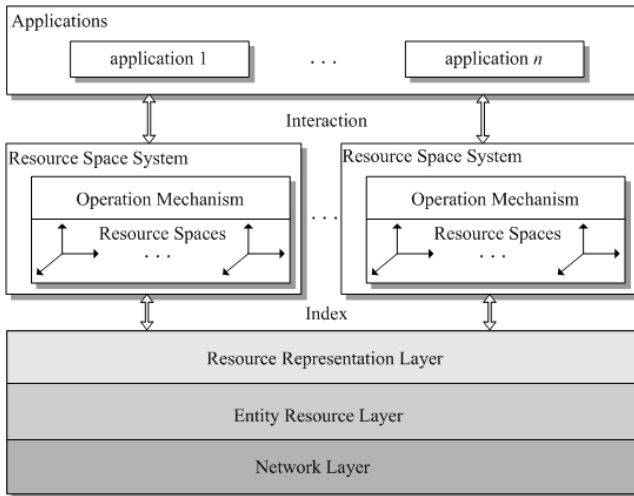


Fig. 1. The architecture of the resource space system

A resource space is an n -dimensional classification semantic space in which every point uniquely determines a set of related resources. The space is denoted as $RS(X_1, X_2, \dots, X_n)$ or just by name RS in simple. $X_i = \{C_{i1}, C_{i2}, \dots, C_{im}\}$ represents axis X_i with its coordinate set. A point $p(C_{1,j1}, C_{2,j2}, \dots, C_{n,jn})$ is determined by a set of coordinate values at all axes. A point can uniquely determine a resource set, where each element is called a *resource entry*. A point can be regarded as the container of a set of resource entries.

Fig. 2 is an example of a two-dimensional resource space *Com-Goods* that specifies goods' information of three companies: Microsoft, Coca Cola and Nike. Two axes in *Com-Goods(Companies, Goods)* are $Companies = \{Microsoft, Coca Cola, Nike\}$ and $Goods = \{soft\ drink, software, dress\}$. Each point specifies a class of goods. For example, the point *(Microsoft, software)* represents the goods belonging to

software and produced by *Microsoft*. A resource entry represents a piece of goods belonging to a point in the *Com-Goods*. Point and resource entry are two types of operation units in RSM.

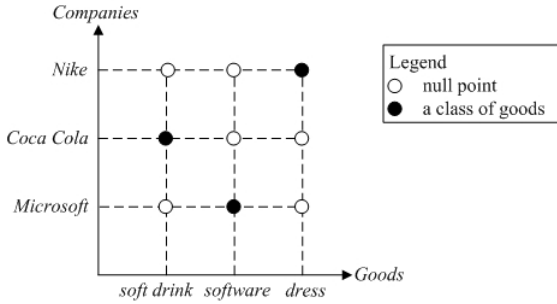


Fig. 2. An example of a two-dimensional resource space

$R(p)$ represents the resources the point p can contain. If $R(p) = \emptyset$, then p is called *null point*, otherwise *non-null point* (the hollow circle and solid circle in Fig. 2). Large number of null points may exist in multidimensional resource spaces. For example, the point (*Microsoft*, *soft drink*) is one of the six null points in the resource space *Com-Goods*. Here focuses on how to identify non-null points.

DEFINITION 1. Let $p.X_i$ be the coordinate of p at axis X_i in $RS(X_1, X_2, \dots, X_n)$, i.e., the projection of p on X_i . If $p_1.X_i = p_2.X_i$ for $1 \leq i \leq n$ holds, then we say that p_1 is equal to p_2 , denoted as $p_1 =_p p_2$.

3 Entity Integrity Constraints

3.1 Key

Keys play fundamental role in the relational data model and its conceptual design. They enable tuples to refer to one another and guarantee operations to accurately locate tuples [1, 4, 14].

As a coordinate system, the RSM naturally supports accurate resource location by giving coordinates. However, it is sometimes unnecessary and even arduous to specify all the coordinates to identify a non-null point, especially for high-dimensional resource spaces. The keys of RSM can further improve the efficiency of resource location.

DEFINITION 2. Let CK be a subset of $\{X_1, X_2, \dots, X_n\}$, p_1 and p_2 be two non-null points in $RS(X_1, X_2, \dots, X_n)$. CK is called a *candidate key* of RS if we can derive $p_1 =_p p_2$ from $p_1.X_i = p_2.X_i, X_i \in CK$.

Candidate keys provide us with necessary and enough information to identify non-null points of a given resource space. Take Fig. 2 for example, $\{Companies\}$, $\{Goods\}$ and $\{Companies, Goods\}$ are candidate keys of the resource space *Com-Goods*. Knowing the coordinates either on $\{Companies\}$ or on $\{Goods\}$, we can eas-

ily get the non-null points as needed. The *primary key* is one of the candidate keys chosen by resource space designers. Any axis belonging to the primary key is called *primary axis*.

Rule 1 (Point constraint): If axis X is one of the primary axes of the resource space RS , then for any point in RS , its coordinate on axis X should not be null.

Rule 1 is used to guarantee the primary keys' functionality of distinguishing non-null points in a given resource space. One type of null value is "at present unknown". Let $\{Companies\}$ be the primary key of *Com-Goods*, the axis *Companies* is a primary axis of *Com-Goods*. If the coordinates of points at axis *Companies* are null, then the points (null, *soft drink*) and (null, *software*) cannot be distinguished by the primary key $\{Companies\}$.

In RSM, some keys can be inferred from the presence of others. This is important in query optimization, especially when dynamically creating resource spaces. Inference rules for candidate keys include the following four theorems.

THEOREM 1. If a set of axes CK is a candidate key of the resource space RS , then any axis set that includes CK is also the candidate key of RS .

Basic operations on resource spaces have been proposed in [18]. Here we introduce relevant definitions (definition 3-6) to develop the theory of integrity constraint.

DEFINITION 3 (Merge of axes). If two axes $X_1 = \{C_{11}, C_{12}, \dots, C_{1n}\}$ and $X_2 = \{C_{21}, C_{22}, \dots, C_{2m}\}$ have the same axis name but have different coordinates, then they can be *merged* into one: $X = X_1 \cup X_2 = \{C_{11}, C_{12}, \dots, C_{1n}, C_{21}, C_{22}, \dots, C_{2m}\}$.

DEFINITION 4 (Join operation). Let $|RS|$ be the number of dimensions of RS . If two resource spaces RS_1 and RS_2 store the same type of resources and they have n ($n \geq 1$) common axes, then they can be *joined* together as one RS such that RS_1 and RS_2 share these n common axes and $|RS| = |RS_1| + |RS_2| - n$. RS is called the join of RS_1 and RS_2 , denoted as $RS_1 \cdot RS_2 \Rightarrow RS$.

DEFINITION 5 (Merge operation). If two resource spaces RS_1 and RS_2 store the same type of resources and satisfy: (1) $|RS_1| = |RS_2| = n$; and (2) they have $n - 1$ common axes, and there exist two different axes X_1 and X_2 satisfying the merge condition, then they can be *merged* into one RS by retaining the $n - 1$ common axes and adding a new axis $X_1 \cup X_2$. RS is called the merge of RS_1 and RS_2 , denoted as $RS_1 \cup RS_2 \Rightarrow RS$, and $|RS| = n$.

DEFINITION 6 (Split operation). A resource space RS can be *split* into two resource spaces RS_1 and RS_2 that store the same type of resources as that of RS and have $|RS| - 1$ common axes by splitting an axis X into two: X' and X'' , such that $X = X' \cup X''$. This split operation is denoted as $RS \Rightarrow RS_1 \cup RS_2$.

According to the definition of the join operation, we have the following theorem.

THEOREM 2. Let RS_1 and RS_2 be two resource spaces that can be joined together to generate a new resource space RS . Assume that CK_1 and CK_2 are the candidate keys of RS_1 and RS_2 respectively. Then, $CK = CK_1 \cup CK_2$ is a candidate key of RS .

Proof. Let A be the set of all axes of RS and A_1 be the set of all axes of RS_1 . We assume that $CK = CK_1 \cup CK_2$ is not the candidate key of RS . Thus, there must exist two

non-null points p_1 and p_2 in RS , which satisfy both $(\forall X \in CK) (p_1.X = p_2.X)$ and $(\exists X^* \in A) (p_1.X^* \neq p_2.X^*)$. Without losing generality, we suppose that the axis X^* exists in RS_1 . It is obvious that $X^* \notin CK_1$. Let p_1' be a point in RS_1 which satisfies $(\forall X \in A_1) (p_1'.X = p_1.X)$ and p_2' be a point in RS_1 which satisfies $(\forall X \in A_1) (p_2'.X = p_2.X)$. According to the definition of *join*, we have $R(p_1) \subseteq R(p_1')$ and $R(p_2) \subseteq R(p_2')$ hold. So, both p_1' and p_2' are non-null points of RS_1 . We can conclude that $(\forall X \in CK_1) (p_1'.X = p_2'.X)$ and $p_1'.X^* \neq p_2'.X^*$. This conclusion obviously contradicts the above assumption that CK_1 is the candidate key of RS_1 . So $CK = CK_1 \cup CK_2$ is a candidate key of RS . \square

The following theorem can be drawn from the definition of the merge operation.

THEOREM 3. Let RS_1 and RS_2 be two resource spaces that can be merged into one resource space RS . Let X_1 and X_2 be two different axes of RS_1 and RS_2 respectively, and let $X_c = X_1 \cup X_2$. Assume that CK_1 and CK_2 are the candidate keys of RS_1 and RS_2 respectively. Then, $CK = (CK_1 - \{X_1\}) \cup (CK_2 - \{X_2\}) \cup \{X_c\}$ is a candidate key of RS .

Proof. Let A be the set of all axes of RS . We assume that $CK = (CK_1 - \{X_1\}) \cup (CK_2 - X_2) \cup \{X_c\}$ is not the candidate key of RS . Thus, there must exist two non-null points p_1 and p_2 in RS , which satisfy both $(\forall X \in CK) (p_1.X = p_2.X)$ and $(\exists X^* \in A) (p_1.X^* \neq p_2.X^*)$. It is obvious that $X^* \notin CK$. Let p_1' and p_2' be two points in RS_1 or RS_2 , which satisfy $(\forall X \in A - \{X_c\}) (p_1'.X = p_1.X)$, $(\forall X \in A - \{X_c\}) (p_2'.X = p_2.X)$, $p_1'.X_1 = p_1.X_c$ (or $p_1'.X_2 = p_1.X_c$), and $p_2'.X_1 = p_2.X_c$ (or $p_2'.X_2 = p_2.X_c$).

Suppose that p_1' and p_2' belong to the same resource space, for example RS_1 . Since $(\forall X \in A - \{X_c\}) (p_1'.X = p_1.X)$, $(\forall X \in A - \{X_c\}) (p_2'.X = p_2.X)$, $p_1'.X_1 = p_1.X_c$, and $p_2'.X_1 = p_2.X_c$ hold, we can reach $(\forall X \in CK_1) (p_1'.X = p_2'.X)$ and $p_1'.X^* \neq p_2'.X^*$. This conclusion obviously contradicts the above assumption that CK_1 is the candidate key of RS_1 .

Suppose $p_1' \in RS_1$ and $p_2' \in RS_2$, and let p_2'' be the point in RS_1 that has the same coordinate values as p_2' . Thus, p_1' and p_2'' have the same coordinate values as p_1 and p_2 respectively. We can reach that $(\forall X \in CK_1) (p_1'.X = p_2''.X)$ and $p_1'.X^* \neq p_2''.X^*$. This obviously contradicts above assumption that CK_1 is the candidate key of RS_1 .

According to (1) and (2), $CK = (CK_1 - \{X_1\}) \cup (CK_2 - X_2) \cup \{X_c\}$ is a candidate key of RS . \square

According to the definition of the split operation, we have the following theorem.

THEOREM 4. Let RS_1 and RS_2 be two resource spaces created by splitting the resource space RS . Suppose that the axis X_c of RS is split into X_1 and X_2 belonging to RS_1 and RS_2 respectively. Let CK be a candidate key of RS . If $X_c \notin CK$ holds, let $CK_1 = CK_2 = CK$, otherwise let $CK_1 = CK - \{X_c\} \cup \{X_1\}$ and $CK_2 = CK - \{X_c\} \cup \{X_2\}$. Then, CK_1 and CK_2 are the candidate keys of RS_1 and RS_2 respectively.

Proof. Let A be the set of all axes of RS and A_1 be the set of all axes of RS_1 . We assume that CK_1 is not the candidate key of RS_1 . Thus, there must exist two non-null points p_1 and p_2 in RS_1 , which satisfy both $(\forall X \in CK_1) (p_1.X = p_2.X)$ and $(\exists X^* \in A_1) (p_1.X^* \neq p_2.X^*)$. Let p_1' and p_2' in RS have the same coordinate values as p_1 and p_2 respectively. It is obvious that $(\forall X \in CK) (p_1'.X = p_2'.X)$ holds in case $CK_1 = CK$ or $CK_1 = CK - \{X_c\} \cup \{X_1\}$ holds.

When $CK_1 = CK$, if $X^* \neq X_1$, then $p_1'.X^* \neq p_2'.X^*$ holds, otherwise $p_1'.X_c \neq p_2'.X_c$ holds;

When $CK_1 = CK - \{X_c\} \cup \{X_1\}$, then $X^* \neq X_1$ holds. So $p_1'.X^* \neq p_2'.X^*$ holds.

According to (1) and (2), $p_1' =_p p_2'$ does not hold. This conclusion obviously contradicts the above assumption that CK is the candidate key of RS . So CK_1 is a candidate key of RS_1 . Similarly, we can prove that CK_2 is a candidate key of RS_2 . \square

In resource space systems, there often exist some resource spaces created dynamically by join, merge and split operations. Theorem 2, 3 and 4 in fact provide efficient means of deriving the candidate keys of these resource spaces created dynamically.

3.2 Resource Entry

In RSM, a resource entry denoted as a 3-tuple *Resource-Entry* $\langle ID, Index, Semantic-Description \rangle$ is used to index a piece of resource of resource representation layer. The first field *ID* is used to specify the resource entries in a given point. Two resource entries residing in different points could have the same ID. The second field *Index* is the index information linked to the representation layer. To facilitate semantic operations, the *Semantic-Description* uses a set of resource attributes to reflect the simple semantics of the resource in the given resource space. The resource representation layer describes the detailed semantics of all resources. In the following discussion, *re.ID*, *re.index* and *re.SD* denote the *ID*, *Index* and *Semantic-Description* of resource entry *re* respectively.

Three types of entity integrity constraints for resource entries are presented as follows. All resource space systems are required to satisfy the first two rules, and the third one is optional according to application requirement.

Rule 2 (Resource entry constraint 1): Any ID should not be null, and for any two resource entries re_1 and re_2 in the same non-null point, $re_1.ID \neq re_2.ID$ holds.

Rule 2 requires that all resource entries in a given non-null point should be distinguishable through IDs. This is helpful to guarantee that any operation can precisely locate the target resource entry.

Rule 3 (Resource entry constraint 2). Any index of a resource entry should not be null, and for any two resource entries in the same non-null point re_1 and re_2 , $re_1.index \neq re_2.index$.

Rule 3 requires the following two conditions: (1) any resource entry should include the index information linked to the representation layer, and (2) there should not exist two different resource entries that have identical index information in a given non-null point. Otherwise, it will lead to information redundancy and unnecessary maintenance of consistency between resource entries in a given point.

The syntactic structure of index information of resource entries depends on the implementation of resource representation layer. For instance, the XML-based implementation of resource representation layer usually uses XPath expressions [5], whereas filenames are often employed for the file-based implementation. To analyze the indexes of resource entries, not only the syntactic structure but also the semantics should be considered. For example, the absolute path differs from the relative path syntactically. However, these two types of paths may represent the same index information.

Rule 4 (Resource entry constraint 3): Any semantic description SD of resource entry should not be null, and for any two resource entries re_1 and re_2 in the same non-null point, they are not the same and do not imply each other in semantics (i.e., neither $re_1.SD \Rightarrow re_2.SD$ nor $re_2.SD \Rightarrow re_1.SD$).

Rule 4 is the entity integrity constraint about the *Semantic-Description* of a resource entry. It is optional and stricter than Rule 3. Since $re.SD$ determines the semantic existence of the resource entries in the resource space, Rule 4 requires that $re.SD$ should not be null. Furthermore, resource entries in a given non-null point should not be the same or imply each other in semantics. For example, a resource and its copies are allowed to coexist in a given non-null point by Rule 3, but not by Rule 4.

4 Membership Integrity Constraint

In relational databases, a tuple can be inserted into a table only if all fields of the tuple satisfy the corresponding domain constraints. So the membership between the tuple and the table should be judged before operation. In RSM, a resource space represents the classification semantics of resources. The existence of resource entry re in point p means that the resource indexed by re belongs to the type represented by p . A resource entry can be inserted into a point by the following operation statement:

INSERT $re\langle ID, index, Semantic-Description \rangle$ **INTO** $p(C_{1,i1}, C_{2,i2}, \dots, C_{n,in})$.

Without any restriction, even a resource entry representing a car could be inserted into the point (*Microsoft, Software*) of the resource space shown in Fig. 2. So, checking the memberships of resource entries plays an important role in RSM.

Note that $R_\Delta(\mathbf{RS})$, $R_\Delta(C)$ and $R_\Delta(p)$ denote the sets of resources currently stored by resource space \mathbf{RS} , coordinate C and point p respectively. For any resource entry re , if re has been inserted into the point p , then $re \in R_\Delta(p)$.

Rule 5 (Membership constraint): Let $re\langle ID, index, Semantic-Description \rangle$ be a resource entry. For any point $p(C_{1,i1}, C_{2,i2}, \dots, C_{n,m})$ in a given resource space, $re \in R_\Delta(p) \rightarrow re \in R(p)$ holds.

As illustrated in Rule 5, a resource entry re can be inserted into point p only if re really belongs to the type that p represents. This membership integrity constraint can avoid incorrect resource classification. When the insert operation or update operation on resource entries is involved, this constraint should be verified.

5 Referential Integrity Constraints

Relational database applications often require that a value appeared in one relation for given set of attributes should also appear for a certain set of attributes in another relation. This condition is called referential integrity constraint. In the following, three types of referential integrity constraints for the RSM are given.

5.1 Referential Integrity for Resource Entries

In RSM, the basic function of a resource entry is to index a certain resource in resource representation layer. For any resource entry $re\langle ID, index, Semantic-$

Description>, *re.index* represents the index information of the corresponding resource. In section 3.2, Rule 3 ensures that *re.index* is non-null. But it cannot guarantee that *re.index* makes sense. This is mainly because some modifications on resource entries or resource representation layer may cause indexes of resource entries to be dead links. The following referential integrity constraint is to eliminate dead links.

Rule 6 (Referential constraint 1). For any resource entry *re* in a given resource space system, there exists a resource in the resource representation layer which is referred by the index information (*re.index*).

The resource space layer is the referencing layer and the resource representation layer is the referenced layer. Rule 6 guarantees that *re.index* makes sense for any resource entry *re*. This constraint should be checked when the insertion of *re* or the update of *re.index*. When delete operation takes place in the resource representation layer, this integrity also needs to be checked.

5.2 Referential Integrity Between Resource Spaces

The first type of referential integrity constraint between resource spaces is relevant to the join operation. In Fig. 3(a), the two-dimensional resource space *Price-Com* stores the same type of resources as the resource space *Com-Goods* (see Fig. 2.). The two resource spaces have a common axis *Companies*, so they can be joined together. Fig. 3(b) depicts the resource space *Com-Goods-Price* created by joining *Price-Com* and *Com-Goods*.

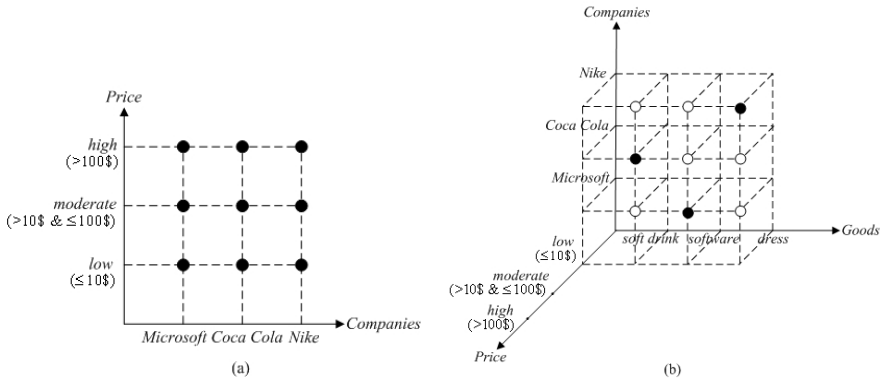


Fig. 3. A two-dimensional resource space and a three-dimensional resource space

All resource entries of *Com-Goods-Price* come from *Price-Com* and *Com-Goods*. *Com-Goods-Price* provides further classification for these resource entries. With no doubt, there exists certain referential relation among *Com-Goods-Price*, *Price-Com* and *Com-Goods*.

Rule 7 (Referential constraint 2). Let RS_1 , RS_2 and RS be three resource spaces that satisfy $RS_1 \cdot RS_2 \Rightarrow RS$, then $R_\Delta(RS) \subseteq R_\Delta(RS_1) \cup R_\Delta(RS_2)$ must hold.

RS is derived from RS_1 and RS_2 , and Rule 7 maintains the dependency of RS on RS_1 and RS_2 . Thus, when a resource entry is inserted into RS or deleted from RS_1 or RS_2 , this constraint should be checked.

The second type of referential integrity constraint between resource spaces assumes that the involved resource spaces satisfy the third-normal-form. We first define the foreign key for RSM.

DEFINITION 7. Let S be a subset of axes of the resource space RS_1 , and it is not the primary key of RS_1 . If there exists another resource space RS_2 such that $R(RS_1) \subseteq R(RS_2)$ holds and S is the primary key of RS_2 , then S is called the *foreign key* of the resource space RS_1 . RS_1 is called the referencing resource space of RS_2 and RS_2 is called the referenced resource space of RS_1 .

According to definition 7, we have the following theorem.

THEOREM 5. Let $S = \{X_1, X_2, \dots, X_m\}$ be the foreign key of the referencing resource space $RS_1(X_1, X_2, \dots, X_m, X_{m+1}, \dots, X_n)$, and $RS_2(X_1, X_2, \dots, X_m, Y_{m+1}, \dots, Y_l)$ be the corresponding referenced resource space. For two non-null points $p(C_1, C_2, \dots, C_m, C_{m+1}, \dots, C_n)$ and $p'(C_1, C_2, \dots, C_m, C'_{m+1}, \dots, C'_l)$ in RS_1 and RS_2 respectively, $R(p) \subseteq R(p')$ holds.

Proof. It is obvious that $R(p') = R(C_1) \cap R(C_2) \cap \dots \cap R(C_m) \cap R(C'_{m+1}) \cap \dots \cap R(C'_l)$. Because RS_2 satisfies the third-normal-form [18, 19] and S is the primary key of RS_2 , $R(p') = R(C_1) \cap R(C_2) \cap \dots \cap R(C_m)$ holds. Since $R(p) = R(C_1) \cap R(C_2) \cap \dots \cap R(C_m) \cap R(C_{m+1}) \cap \dots \cap R(C_n)$ holds, we have $R(p) \subseteq R(C_1) \cap R(C_2) \cap \dots \cap R(C_m)$. So $R(p) \subseteq R(p')$ holds.

The theorem 5 indicates the inclusion relation between the points in the referencing resource space and their counterparts in the referenced resource space. The following constraint is to maintain the legal referential relation between the referencing resource space and its referenced resource space.

Rule 8 (Referential constraint 3). Let $S = \{X_1, X_2, \dots, X_m\}$ be the foreign key of the referencing resource space $RS_1(X_1, X_2, \dots, X_m, X_{m+1}, \dots, X_n)$, and $RS_2(X_1, X_2, \dots, X_m, Y_{m+1}, \dots, Y_l)$ be the corresponding referenced resource space. For two non-null points $p(C_1, C_2, \dots, C_m, C_{m+1}, \dots, C_n)$ and $p'(C_1, C_2, \dots, C_m, C'_{m+1}, \dots, C'_l)$ in RS_1 and RS_2 respectively, $R_\Delta(p) \subseteq R_\Delta(p')$ holds.

Rule 8 guarantees that if a resource entry re appears in a point p in the referencing resource space, then re must exist in the counterpart of p in the referenced resource space.

6 User-Defined Integrity Constraints

Any resource space systems should conform to entity integrity constraints, membership integrity constraints and referential integrity constraints. In specific applications, different resource space systems should obey different context-relevant constraints. These constraints are called user-defined integrity constraints. Here introduces three frequently used types of user-defined constraints. Two resource spaces shown in Fig. 4 are used to illustrate the user-defined and application-relevant integrity constraints. In Fig. 4(a), the resource space *Age-Gender* is used to accommodate employees' information. Every point classifies these employees by their age and gender. In Fig. 4(b), resource space *Tutor-Class* is used to represent students' information. Each point of *Tutor-Class* classifies these students by their tutor and class information.

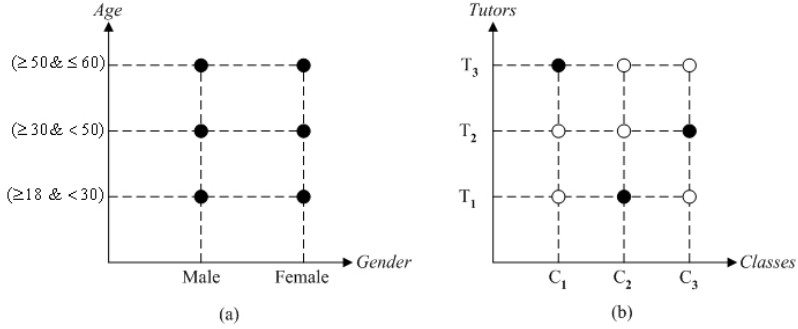


Fig. 4. Examples of two-dimensional resource space

6.1 Value-Based Constraint

This type of user-defined constraints requires the attributes' values in resource description to satisfy some rules. Function $GetValue(attr)$ returns the value of attribute $attr$ specified in the *Semantic-Description* of a certain resource entry. This type of constraints can be described as follows:

$\langle Constraint\ expression \rangle ::= \langle Operand \rangle \langle Relation-Op \rangle \langle Operand \rangle \mid$
 $\langle Constraint\ expression \rangle \vee \langle Constraint\ expression \rangle \mid$
 $\langle Constraint\ expression \rangle \wedge \langle Constraint\ expression \rangle \mid$
 $\neg \langle Constraint\ expression \rangle;$
 $\langle Operand \rangle ::= GetValue(attr) \mid user-defined-constant-value;$
 $\langle Relation-Op \rangle ::= < | > | = | \leq | \geq | \neq.$

Take Fig. 4(a) for example, if the designer requires that any male employee should not be older than 70 and female should not be older than 60 in *Age-Gender*, then this user-defined constraint can be described as follows:

For any resource entry re ,

$(GetAttribute(gender) = \text{"male"} \wedge GetAttribute(age) \leq 60) \vee$
 $(GetAttribute(gender) = \text{"female"} \wedge GetAttribute(age) \leq 55)$ holds.

Before the resource entry re can be inserted into *Age-Gender* or updated, the system should check whether the above constraint has been violated.

6.2 Resource-Entry-Based Constraint

In some applications, semantic relations among resource entries should be considered. Operations on a resource entry may require other operations on semantically relevant resource entries. This type of user-defined constraints is called resource-entry-based constraint. For example, RS is a resource space that contains all registration information of students in a school and RS' is a resource space that contains all health information of these students. Let re be the resource entry representing a student's registration information and re' be the resource entry representing his/her health information. The health information depends on the valid registration information, i.e., $re' \in R_{\Delta}(RS') \rightarrow re \in R_{\Delta}(RS)$ must hold. So this constraint should be checked before the insertion of re' or after the deletion of re .

6.3 Point-Based Constraints

As resource sets, points are often required to satisfy some application relevant rules from the viewpoint of set theory. Take Fig. 4(b) for example, suppose that a class has only one tutor in *Tutor-Class* and that each tutor is in charge of only one class. For any T_i , there at most exists one C_j such that $R_\Delta(\mathbf{p}(T_i, C_j)) \neq \emptyset$, and for any C_m there at most exists one T_n such that $R_\Delta(\mathbf{p}(T_n, C_m)) \neq \emptyset$. We define function $NotNull(\mathbf{p}) = \begin{cases} 1, & R_\Delta(\mathbf{p}) \neq \emptyset \\ 0, & R_\Delta(\mathbf{p}) = \emptyset \end{cases}$ and use \mathbf{p}_{ij} to denote the point $\mathbf{p}(T_i, C_j)$. Then, this constraint can be formally represented as:

$\forall i (\sum_{j=1}^3 NotNull(\mathbf{p}_{ij}) \leq 1) \wedge$
 $\forall j (\sum_{i=1}^3 NotNull(\mathbf{p}_{ij}) \leq 1)$.

Thus, before any student information can be inserted into *Tutor-Class*, the system needs to check whether the above constraint is violated or not.

7 Implementation

A resource space system includes functions for resource space definition, manipulation and system management. Its underlying metadata and data structure use XML and XML schema as shown in Fig. 5. An xml schema *rsm-schema* given at kg.ict.ac.cn/rsm/rsm-schema.xsd specifies the generic definitions of resource space such as *resource space*, *axis*, *coordinate*, and *integrity constraint*. The domain resource space schema is the derivation of the *rsm-schema*. Resource spaces can be specified by the XML files, which are regulated by domain specific schemas. The following statement defines resource spaces:

```

create resource_space rsname (
    (axis_name(coord_name[, coord_name]...)
    [, axis_name(coord_name[, coord_name]...) ]...)
    primary key (axis_name[, axis_name]...)
    [no_sem_duplication]
    [[foreign key (axis_name[, axis_name]...) references rsname]...]
    [join_ref (rsname[, rsname]...)]
    [[check expression]...]
)

```

The users and applications can define the schemas of resource spaces and specify which optional constraints these resource spaces should obey besides the required ones. The clauses “*no_sem_duplication*”, “*join_ref (...)*”, “*foreign key (...)* *reference...*”, and “*check...*” represent that the target resource space should comply with the entity constraint on entry semantic description, the Join operation relevant reference constraints, the foreign key based reference constraints and the user-defined

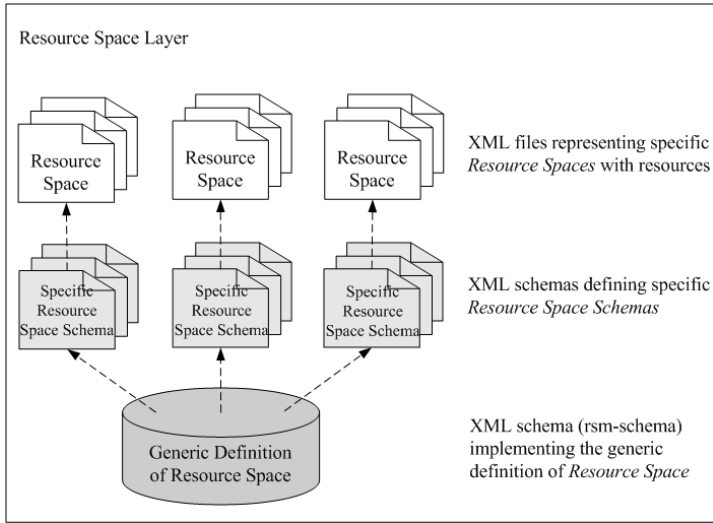


Fig. 5. The implementation of the Resource Space

constraints respectively. Here demonstrates how the integrity constraints of RSM are expressed in the *rsm-schema* of the resource space system.

Point Constraint. The following XML definition defines the axes of RSM. The statement “`<xs:attribute name="iskey" type="xs:boolean"/>`” defines a flag for each axis from which the resource space system can judge whether a certain axis is one of the primary axes of a resource space. Therefore, the resource space system can determine the primary key of a given resource space.

```

<xs:complexType name="rsm:Axis">
  <xs:sequence>
    <xs:element ref="rsm:coordHierarchy" maxOccurs="unbounded"/>
    <!-- rsm:coordHierarchy represents the definition of coordinate hierarchy -->
  </xs:sequence>
  <xs:attribute name="axname" type="xs:string" use="required"/>
  <xs:attribute name="iskey" type="xs:boolean"/>
</xs:complexType>

```

Resource Entry Constraint. The first part of the following definition is the entry in *rsm-schema*: *<ID, Index, Semantic-Description>*. In the second part, each point of a resource space consists of a number of resource entries. The code in bold has defined two types of “*xs:key*” in “*point*”. Therefore, the resource space system makes use of the feature of “*xs:key*” in XML schema to guarantee the uniqueness of *ID* and *Index* of resource entries respectively. Since resource entry constraint 3 is optional, the third part has defined a boolean variable “*IC-sem-entry*” through which the resource space system can determine whether a resource space need to keep this semantic description constraint.


```

<xs:complexType name="Entry">
  <xs:sequence>
    <xs:element ref="rsm:entryID"/>
    <xs:element ref="rsm:entryIndex"/>
    <xs:element ref="rsm:entrySD"/>
  </xs:sequence>
</xs:complexType>
.....
<xs:complexType name="Point">
  <xs:sequence>
    <xs:element ref="rsm:axes"/>
    <xs:element ref="rsm:entry" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="point" type="rsm:Point">
  <xs:key name="KeyOfEntryID">
    <xs:selector xpath="rsm:entry"/>
    <xs:field xpath="rsm:entryID"/>
  </xs:key>
  <xs:key name="KeyOfEntryIndex">
    <xs:selector xpath="rsm:entry"/>
    <xs:field xpath="rsm:entryIndex"/>
  </xs:key>
</xs:element>
.....
<xs:simpleType name="IC-sem-entry">
  <xs:restriction base="xs:boolean"/>
</xs:simpleType>

```

Membership Constraint guarantees a resource entry to fall into the proper point in a resource space. Although there is no corresponding code to reflect the membership constraint in the *rsm-schema*, the resource space system can provide a mechanism to make sure that all resource spaces obey this constraint. Before a new resource entry is inserted into a point or after an existing resource entry of a point is updated, the resource space system needs to retrieve the corresponding features of the resource over the point to check whether the resource belongs to this point. If the membership constraint is violated, the insert or update operation should be cancelled.

Reference Constraint. The reference constraint 1 requests that the resource entity corresponding to the index of a resource entry in a resource space must be represented in the resource representation layer. Once modification to URIs of resource entities in the representation layer or to the indices of resource entries takes place, the resource space system needs to check whether the reference constraint 1 is violated.

As for the reference constraints 2 and 3, using the code in bold in the first part of the following XML schema definition, the resource space system can record the reference relationship between resource spaces. Thus, once resource entries are inserted into referring resource spaces or removed from referred resource spaces, the resource space system will check whether the referring resource spaces and the referred resource spaces are compatible with the reference relationship.

```

<xs:simpleType name="RefSpace">
  <xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<xs:complexType name="IC-ref-join">
  <xs:sequence>
    <xs:element name="joiningSpace" type="rsm:RefSpace"/>
    <xs:element name="joiningSpace" type="rsm:RefSpace"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="IC-foreign">
  <xs:sequence>
    <xs:element name="foreignSpace" type="rsm:RefSpace"/>
  </xs:sequence>
</xs:complexType>
.....
<xs:complexType name="IC-userdefined">
  <xs:sequence>
    <xs:element ref="rsm:check" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

```

User-defined Constraint. In the second part of above XML schema definition, the resource space system uses “*rsm:check*” elements to record all the user-defined constraints a resource space should comply with. Once any of these constraints is broken, the resource space system should cease the operations. The following is the definition of “*Constraint*” element in *rsm-schema*.

```

<xs:complexType name="Constraint">
  <xs:sequence>
    <xs:element name="ic-sem-entry" type="rsm:IC-sem-entry"/>
    <xs:element name="ic-ref-join" type="rsm:IC-ref-join" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element name="ic-foreign" type="rsm:IC-foreign" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element name="ic-userdefined" type="rsm:IC-userdefined" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

```

Where the element “*Constraint*” is defined to totally specify which optional constraints should be complied with besides the required ones. The optional constraints include entity constraint about entry semantic description (“*ic-sem-entry*”), Join operation related reference constraints (“*ic-ref-join*”), foreign key based reference constraints (“*ic-foreign*”), and user-defined constraints (“*ic-userdefined*”).

8 Application in Dunhuang Culture Exhibition

Dunhuang of west China includes over 1000 ancient caves containing precious wall-painting and color statues. Our project Dunhuang Culture Grid is to exhibit the artifacts on the Internet by using the technologies of animation, virtual reality and the

Knowledge Grid [19]. The fundamental work is to establish the resource spaces for the artifacts in caves and to set their integrity constraints. Fig. 6 illustrates the resource space modeling of a cave in contrast to the relational modeling. The resource space is based on classification semantics. A point in a well-defined resource space uniquely determines a class of artifacts.

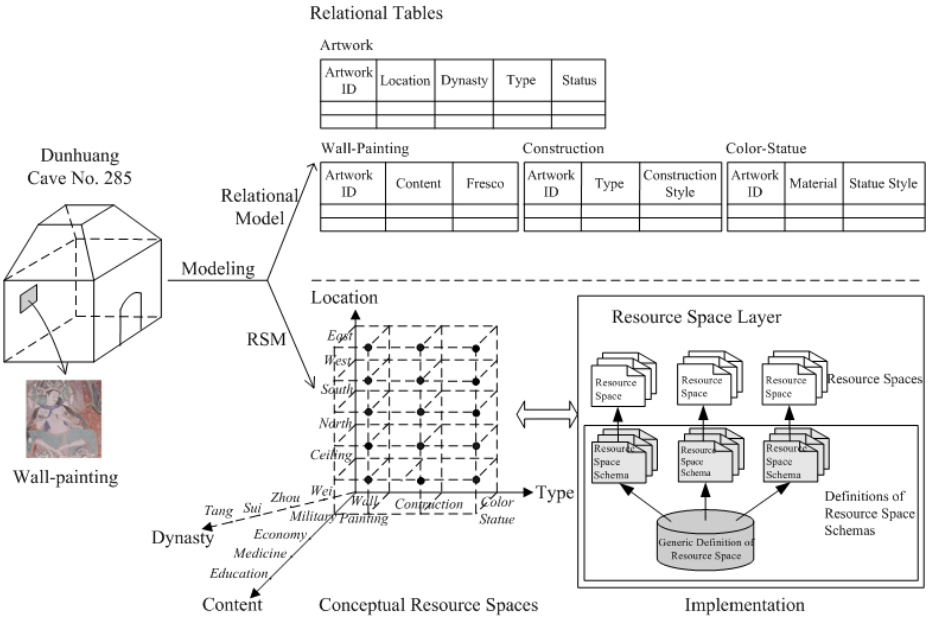


Fig. 6. RSM modeling in contrast to relational modeling

The Dunhuang culture resource space schemas derived from the *rsm-schema* are given at kg.ict.ac.cn/rsm/dh/rsm-dunhuang.xsd. Here uses two resource spaces of the system to illustrate how to define the integrity constraints for resource spaces. The resource space *fresco-285* is to specify all the wall-paintings in cave no. 285. The following is the creation statement:

```

create resource_space fresco-285 (
    (dynasty(Wei, Zhou, Sui, Tang),
    location(east, south, west, north),
    status(spoilt, maintained, available)
    )
    primary key (dynasty, location, status)
    no_sem_duplication
)
    
```

All its dimensions *dynasty*, *location* and *status* constitute the primary key. This resource space also requires that the entity constraint about the entry semantic description needs to be maintained. A resource space *flying-deity-285* described below is to manage the unspoiled flying deities in cave no. 285.

```

create resource_space flying-deity-285 (
  (dynasty(Wei, Zhou, Sui, Tang),
   location(east, south, west, north),
   status(spoilt, maintained, available),
   flying-style(childish, naked, flowered)
  )
  primary key (dynasty, location, status, flying-style)
  no_sem_duplication
  foreign key (dynasty, location, status) references fresco-285
  check status≠'spoilt'
)

```

flying-deity-285 has four dimensions: *dynasty*, *location*, *status* and *flying-style*, they constitute the primary key. The axes *dynasty*, *location* and *status* are the foreign key that results in the reference from *flying-deity-285* to *fresco-285*. A user-defined constraint “*status*≠’*spoilt*’” has been defined to indicate that *flying-deity-285* can only contain the unspoiled flying deities.

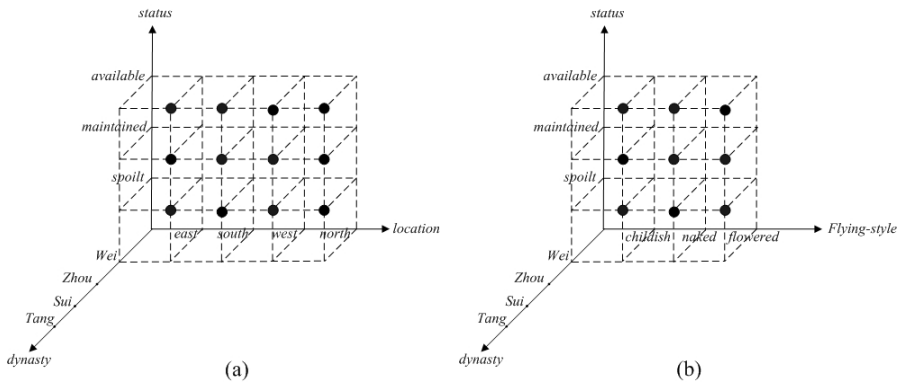


Fig. 7. The resource space *fresco-285* (a) and a slice of the resource space *flying-deity-285*(b)

Fig. 7 (b) corresponds to the *west* coordinate in the axis *location*. The integrated XML files representing *fresco-285* and *flying-deity-285* are available at kg.ict.ac.cn/rsm/dh/fresco-285.xml and kg.ict.ac.cn/rsm/dh/flying-deity-285.xml.

9 Conclusion

Based on our resource space model RSM, this keynote proposes the integrity theory for RSM. It is the basis for guaranteeing the correctness of operations on resource spaces. The implementation approach and application in culture area demonstrate its usability. More background and practice on Dunhuang Cultural Grid and Knowledge Grid are available at www.culturegrid.net and www.knowledgegrid.net. We are investigating the integrity on a new semantic model integrating RSM with SLN (a Semantic Link Network model [19]).

References

1. Abiteboul, S., Hull, R. and Vianu, V.: Foundations of Databases. Addison-Wesley, Reading, MA (1995).
2. Berners-Lee, T., Hendler, J. and Lassila, O.: Semantic Web. *Scientific American*, 284 (5) (2001) 34-43.
3. Bray, T., Paoli, J. and Sperberg-McQueen, C.M.: Extensible Markup Language (XML) 1.0. W3C Recommendation, February 1998. www.w3.org/TR/REC-xml/.
4. Buneman, P. et al.: Keys for XML. In: *Proceedings of International World Wide Web Conference, WWW10*, Hong Kong, 1-5 May 2001.
5. Clark, J. and DeRose, S.: XML Path Language (XPath). W3C Working Draft, November 1999. www.w3c.org/TR/xpath.
6. Codd, E.F.: A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*, 13 (6) (1970) 377-387.
7. Codd, E.F.: Extending the Database Relational Model to Capture More Meaning. *ACM Transactions on Database Systems*, 4 (4) (1979) 397-434.
8. Date, C.J.: Referential Integrity. In: *Proceedings of International Conference on Very Large Data Bases*, Cannes, 9-11 September 1981.
9. Davidson, A. et al.: Schema for Object-Oriented XML 2.0. W3C Note, July 1999. www.w3c.org/TR/NOTE-SOX.
10. Eswaran, K.P. and Chamberlin, D. D.: Functional Specifications of a Subsystem for Database Integrity. In: *Proceedings of International Conference on Very Large Data Bases*, Framingham MA, 22-24 September 1975.
11. Fan, W. Kuper, G. and Simon, J.: A Unified Constraint Model for XML. In: *Proceedings of International World Wide Web Conference, WWW10*, Hong Kong, 1-5 May 2001.
12. Hammer, M.M. and Mcleod, D.J.: Semantic Integrity in a Relational Data Base System. In: *Proceedings of International Conference on Very Large Data Bases*, Framingham MA, 22-24 September 1975.
13. Layman, A. et al.: XML-Data. W3C Note, January 1998. www.w3c.org/TR/1998/NOTE-XML-data.
14. Ramakrishnan, R. and Gehrke, J.: Database Management Systems. McGraw-Hill Higher Education, New York, 2000.
15. Stonebraker, M.: Implementation of Integrity Constraints and Views by Query Modification. In: *Proceedings of ACM-SIGMOD International Conference on the Management of Data*, San Jose CA, 14-16 May 1975.
16. Thompson, H.S. et al.: XML Schema. W3C Working Draft, May 2001. www.w3c.org/XML/Schema.
17. Zhuge, H.: Resource Space Model, Its Design Method and Applications. *Journal of Systems and Software*, 72 (1) (2004) 71-81.
18. Zhuge, H.: Resource Space Grid: Model, Method and Platform. *Concurrency and Computation: Practice and Experience*, 16 (14) (2004) 1385-1413.
19. Zhuge, H.: The Knowledge Grid, *World Scientific*, Singapore, 2004.
20. Zhuge, H., Yao, E., Xing, Y. and Liu, J.: Extended Normal Form Theory of Resource Space Model. *Future Generation Computer Systems*, 21 (2005) 189-198.

Challenges of Grid Computing*

Hai Jin

Cluster and Grid Computing Lab,
School of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, 430074, China
hjin@hust.edu.cn

Abstract. Grid computing, being a promising way for distributed supercomputing from its very beginning, attracts many attentions worldwide. Many national or regional grid computing projects come into attention. Although lots of fruitful results are obtained through such huge projects, grid computing is still far way from what we expect, using all kinds of resources and services over the internet just as using public utilities like water and electricity. There are lots of challenges for grid computing, both from technical point of view and from non-technical point of view. Also, there are quite a few barriers from grid computing being widely accepted by ordinary end users. In this talk, detail discussions of these challenges and barriers for the development and usage of grid computing are listed. The purpose of this talk tries to point out that grid computing still has long way to go to be more accepted not only by scientists but also by ordinary end users.

1 Introduction

Just one day before 911 event, a paper appeared in the Forbes ASAP [23], stated that the second generation of the Internet would happened in 2004 or 2005, and it would become a \$20 trillion industry by the year 2020. The key for Internet II is Great Global Grid instead of World Wide Web. Unlike the current World Wide Web, the Great Global Grid will be primarily a visual medium. The grid will be everywhere in our daily lives--automobile dashboards, wristwatches, PDAs, cell phones, appliances, game boys, cash registers, even on the walls of public spaces.

Actually, the idea of grid computing can be traced back to metacomputing [14]. With the appearance of Globus [13][14], more and more attentions are put to the grid computing [1][3][15][17][21][25][31]. Grid computing visions to have all the resources and services over internet as a utility just like the way we use electricity and water. We need not to know the source of information, just as we do not know the power station. We need not to know the type of machines providing the information, just as we do not know the type of electricity.

There are lots of potential applications of grid computing. Besides traditional distributed supercomputing, such as TeraGrid [35], IPG [26], GIG [22], UK e-Science

* This work is supported by National Science Foundation under grant 60125208, 60273076 and 90412010, ChinaGrid project from Ministry of Education of China, National 863 Hi-Tech R&D Research Program under grant 2004AA104280, and National 973 Basic Research program under grant 2003CB317003.

[38], and ChinaGrid [6][20][39], *computer supported cooperative work* (CSCW), high throughput computing with massive simulation and parameter study, remote low-cost software access or software leasing, data intensive computing, such as European DataGrid project [8], and on-demand computing are all the promising driven forces for grid computing. Grid computing will evolve over the next 10 years into a mainstream IT infrastructure for business applications.

2 Challenges of Grid Computing

Although it seems promising to use grid computing to eliminate the resource islands and to provide transparent way to access resources and services over internet, we till know still could not see the very perspective of using grid computing. Some even states that grid computing rates a 6.5 on a scale of 10 on the hype meter [11]. Till now, we could not have the "dial tone" of the world's aggregating computing power, but the resources for that capability have already existed. But lots of enterprises are able to leverage existing excess computing power through private grids. In maybe five years, we may see massive public grids on controlled basis--meaning inside a VPN or a secure environment [11].

To achieve above goals, we need to analysis the current challenges and barriers in developing, deploying, promoting, and using of grid computing. In this talk, I briefly list the following challenges for possible discussions and researches. This talk are based on my discussions with various peoples, including grid middleware developers, grid administrators, grid application developers, and grid users.

Challenge 1: There is no clear standard to follow

In order to masks the heterogeneous features of different resources in grid environment, standard is the very first thing needs to be worked out. From the very beginning of *Global Grid Forum* (GGF) [12], standard is the most important task for GGF. Till now, *Open Grid Systems Architecture* (OGSA) [29][37] is now accepted by more and more people, more and more voices from industries advocate *Web Services Resource Framework* (WSRF) [7]. Even though, there are still different tones for the future standards of grid computing. Without the widely accepted standards, the more grid applications are developed, the more resource islands will incur.

Challenge 2: Still lots of debate on what grid computing is, and what is not

From the very beginning of grid computing, the definition or the scope of grid computing has been constant changing. One popular the definition from Globus team is that grid computing "enables coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations" [17]. But due to the challenge 4 below, more and more people in grid computing area turn their attention from grid computing to services computing. Also with the success stealing of individual computer cycles, more and more peer-to-peer style computing paradigm are utilized to fulfill the requirement of high throughput computing, such as SETI@Home [33].

Now, more and more debates are raised for what indeed the grid computing is. Maybe the following words from UK Prime Minister Tony Blair are a possible understanding of what the grid computing from ordinary users, grid "intends to make ac-

cess to computing power, scientific data repositories and experimental facilities as easy as the Web makes access to information” [4].

Challenge 3: Grid application development is still difficult

Although there are some efforts working on how to provide users interface for grid computing, the development of grid application is still very difficult and expensive. These interfaces are not designed for grid application developers. Most of grid applications are implemented with the help of computer scientists case by case. This in some senses embarrasses the widely use of grid computing. To provide a MPI-like grid application programming language is not enough, as many scientists are not familiar with parallel programming language, and more and more grid applications are not scientific computing anymore. The ideal way for the grid computing application development is to use drag-and-play fashion. To achieve this goal, a suite of grid application development toolkit is needed.

Challenge 4: Application area is limited and significant applications are lacked

For most scientific applications, more and more super clusters can meet the requirements of applications. For medium size scientific computations, it is much easier and efficient to own and use a local super cluster. For special applications, such as global weather modeling and bioinformatics, a special designed super computer is needed, such as Earth-Simulator [9], Blue-Genes [5].

Due to the network latency and delay, a global super computer is not the most optimized project than others in performance/price ratio. Especially for data processing over the grid, it is much more cost-effective to build a data center other than using dataset remotely, as data storage is inexpensive than data transfer. Services computing [10][19] could be a possible killer application style for the future grid computing, as services are a much easier way to use than to own.

Challenge 5: Lots of efforts should be done to make a software package or a service useable over grid

One major resource for the researchers is existing software package. Many existing software packages are running on a dedicate platform, such as SMP cluster, or over a dedicate operating system. Sometimes, it is very difficult to have them reusable over grid environment due to various reasons, such as lack of source code, copyright issues. Although there are some software vendors put their efforts on this issue, such as Oracle 10g [30], this figure with grid-enable software packages is far behind the development of grid middleware. For the time being, these software packages are encapsulated into a grid services so that requests to these software packages can be re-directed to the grid node providing the services. Although this method solves the problem in some extent, in order to have more collaborative researches over grid, one important effort needed is to have more and more software packages have their grid versions.

Challenge 6: Centralized management of grid computing

Currently, most of grid computing projects are in centralized management scheme. This is due to two main reasons. The first is that for most grid projects, participant organizations or institutions are limited, centralized management can be easily used to

manage all the resources and services within a grid domain. The second is most grid middleware use “publish-find-bind” web services scheme [40]. All the resources and services need to be registered to the *Universal Discovery, Description and Integration* (UDDI) center. This UDDI center becomes the central management point of grid system.

This single domain scheme restricts the scalability of grid entities joining the effort, and also will be the single point of failure of the whole grid system. *ChinaGrid Support Platform* (CGSP) [20] from ChinaGrid project is a first step towards multi-domain web services architecture. Therefore a domain manager is the key design philosophy in CGSP. Each domain manages the resources and services within the domain, and all the domain managers among all the domains work collaboratively to exchange information and scheduling resources.

Challenge 7: Lack of resource sharing between variant types of services

Each type of service has its own standard and protocol. Major grid computing applications only provides one type of service at present. This is because of several reasons. One reason is that most grid applications are developed for different areas. Different grid applications have different focus, sharing computing power, or sharing data/databases. The other reason is that different grid applications using different platforms. The resources or services for a particular grid platform could not be shared by other grid platforms. WSRF is a major effort to standardize the grid platform and the way to encapsulate and publish resources and services among different grid platform. But still, more and more computation oriented or data processing oriented grid applications still doubt the efficiency of using web services for computation and data processing.

Challenge 8: Lack of security/trust between different services

The US *Department of Energy* (DoE) Office of Advanced Scientific Computing Research published a report which provides a good summary of the requirements for grid security [2]. The grid requires a security infrastructure with the following properties: ease of use by users; conformation with the VO security needs while at the same time working well with site policies of each resource provider site; provisions for appropriate authentication and encryption of all interactions.

The *Grid Security Infrastructure* (GSI) [16] and MyProxy [27] are two primarily important works for the grid security. But the two-party authentication protocols of GSI do not provide an adequate solution to group oriented grid security applications. GSI cannot easily achieve a common key for a VO wide encrypted communication. Additionally, they do not have a inherent means for realizing behavior control for a remote user and its client system environment. For example, consider that WS-security [28] can achieve message encryption between a resource provider and a user. However, there is no way for a stakeholder in the resource provider to know whether or not the remote client environment is compromised (perhaps by a malicious code) even though it knows that such a compromise is equivalent to the nullification of the channel encryption service [24]. Trusted computing [36], which is an integrity protection of resources naturally, suits the security requirements for grid computing or science collaborations.

Challenge 9: Business model of grid is ambiguous

On-demand computing is often used as an example of a business model that can be used for grid computing. The model is that computing power is offered by companies or centers with idle computing power to companies that need computing power. This would be one of the first areas, where grid computing would have a major impact on business. Due to the network costs, this is not a good model in general [18]. Only for very compute intensive applications, like rendering, it makes sense. In most other cases, a Beowulf cluster [34], with faster network connections than WANs, is a more inexpensive choice. For 1 US\$ one can get 1 GB sent over the WAN, or 10 Tops tera-CPU instructions, or 8 hours of CPU time, or 1 GB disk space, or 10 M database accesses, or 10 TB of disk bandwidth. It is fine to send a Gbyte of data over the network if it saves years of computation. But it is not economic to send a Kbyte question if the answer could be computed locally in a second. On-Demand computing is only economical for very CPU-intensive applications: about a CPU-day per Gbyte of network traffic.

Challenge 10: Management and administration of grid is the most challenged one

All above challenges are technical issues for grid computing. However, grid computing is not only a technical concern. It is a huge project across many organizations and institutes in a geographically distributed environment. Therefore, management and administration of grid computing is the key to the success of grid project, especially when the design purpose for grid computing project is a production grid. UK e-Science program [38] gives us a good example of how to setup national grid center and regional grid center, as well as some grid functional center, such as grid R&D center, grid support center, grid training center, grid software verification center.

Also, there are huge resources and services existing in the grid environment. All these resources and services needed to be manageable. There are lots of software packages for monitoring the hardware resources, but seldom has functionality of monitoring services over the grid environment. How to monitor and manage these services efficiently and in real-time within grid domain is also a challenge.

3 Conclusion

In this paper, several key issues of the challenges and barriers for grid computing have been addressed. In one word, there is a long way to go for grid computing being widely accepted by end users. To end of this paper, I would like to quote a very interesting passage from [32] to state the metrics for the success of grid computing:

The Grid can be considered a success when there are no more “Grid” papers, but only footnote in the work that states, “This work was achieved using the Grid”.

The Grid can be considered a success when a supercomputer centers do not give a user the choice of using their machines or using the Grid, they just use the Grid.

The Grid can be considered a success when a SuperComputing (SC) demo can be run any time of the year.

Besides the above metrics, I would like to add two more metrics to further extend the success of grid computing from scientific computation area to service computing area:

The Grid can be considered a success when it becomes a common tool to build various applications, and just like webpage, the people using grid to design their applications are not computer scientists.

The Grid can be considered a success when a software package or a service can just be uploaded to the grid and become a grid wide service just like we use memory sticks, although they are produced from different manufactures, they can be immediately recognized by various computers.

References

1. A. Abbas, *Grid Computing: A Practical Guide to Technology and Applications*, Charles River Media, 2003.
2. D. Agarwal, R. Bair, et. al., *National Collaboratories Horizons*, Report of the August 10-12, 2004, National Collaboratories Program Meeting, the U.S. Department of Energy Office of Science.
3. F. Berman, G. Fox, and A. J. G. Hey (eds), *Grid Computing: Making The Global Infrastructure a Reality*, John Wiley & Sons, 2003.
4. Tony Blair Speech, *Science matters*, <http://www.number-10.gov.uk/output/Page1715.asp>, April 10, 2002.
5. The Blue Gene Project, <http://www.research.ibm.com/bluegene/>.
6. ChinaGrid, <http://www.chinagrid.edu.cn>.
7. K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe, *The WS-Resource Framework*, <http://www.globus.org/wsrff/>
8. Data Grid Project WP1, "Definition of Architecture, Technical Plan and Evaluation Criteria for Scheduling, Resource Management, Security and Job Description", *Datagrid document DataGrid-01-D1.2-0112-0-3*, 14/09/2001.
9. The Earth Simulator Center, <http://www.es.jamstec.go.jp/>.
10. T. Erl, *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*, Prentice Hall PTR, 2004.
11. D. Farber, "Grid computing rates a 6.5 on the hype meter", *Tech Update*, May 15, 2002.
12. Global Grid Forum, <http://www.ggf.org>.
13. Globus, <http://www.globus.org>.
14. I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", *International Journal of Supercomputer Applications*, Vol.11, No.2, pp.115-128, 1997.
15. I. Foster and C. Kesselman (eds.), *The Grid 2: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, 2003.
16. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, "A security architecture for Computational Grids", *Proceedings of 5th ACM Conference on Computer and Communications Security*, pp.83-92, 1998.
17. I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International Journal of High Performance Computing Applications*, 15 (3), 200-222, 2001.
18. J. Gray, "Distributed Computing Economics", *GRIDtoday*, Vol.2, No.29, July 21, 2003.
19. IEEE Services Computing Community, <https://www.ieeecommunities.org/services>.
20. H. Jin, "ChinaGrid: Making Grid Computing a Reality", *Digital Libraries: International Collaboration and Cross-Fertilization - Lecture Notes in Computer Science*, Vol.3334, Springer-Verlag, December 2004, pp.13-24.
21. J. Joseph and C. Fellenstein, *Grid Computing*, Prentice Hall PTR, 2003.

22. M. Libicki, *Who Runs What in the Global Information Grid: Ways to Share Local and Global Responsibility*, RAND, 2000.
23. M. S. Malone, "Internet II: Rebooting America", *Forbes ASAP*, Sept. 10, 2001.
24. W. Mao, "Innovations for the Grid Security from the Trusted Computing", *Technical Report*, Hewlett-Packard Laboratories, Feb. 2005.
25. D. Minoli, *A Networking Approach to Grid Computing*, Wiley-Interscience, 2004.
26. NASA Information Power Grid, <http://www.ipg.nasa.org>.
27. J. Novotny, S. Teucke, and V. Welch, "An Online Credential Repository for the Grid: MyProxy", *Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10)*, August 2001.
28. M. O'Neill, *Web Services Security*, McGraw-Hill Osborne Media, 2003.
29. Open Grid Services Architecture, http://www.ggf.org/Public_Comment_Docs/Documents/draft-ggf-ogsa-specv1.pdf.
30. Oracle 10g, <http://www.oracle.com/database/index.html>.
31. P. Plaszczak and Jr. R. Wellner, *Grid Computing: The Savvy Manager's Guide*, Morgan Kaufmann, 2005.
32. J. M. Schopf and B. Nitzberg, "Grid: The Top Ten Questions", *Scientific Programming*, Special Issue on Grid Computing, 10(2):103-111, August 2002.
33. SETI@Home, <http://setiathome.ssl.berkeley.edu/>.
34. T. Sterling, J. Salmon, D. J. Becker, and D. F. Savarese, *How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters*, MIT Press, 1998.
35. The TeraGrid Project, <http://www.teragrid.org/>.
36. Trusted Computing Group, <https://www.trustedcomputinggroup.org/>.
37. S. Tuecke, Kzajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, D. Snelling, and P. Vanderbilt, *Open Grid Services Infrastructure*, February 17, 2003.
38. UK e-Science Programme, <http://www.rcuk.ac.uk/escience/>.
39. G. Yang, H. Jin, M. Li, N. Xiao, W. Li, Z. Wu, Y. Wu, and F. Tang, "Grid Computing in China", *Journal of Grid Computing*, Vol.2, No.2, June 2004, pp.193-206.
40. O. Zimmermann, M. R. Tomlinson, and S. Peuser, *Perspectives on Web Services: Applying SOAP, WSDL and UDDI to Real-World Projects*, Springer, 2005.

BBTC: A New Update-Supporting Coding Scheme for XML Documents

Jianhua Feng, Guoliang Li, Lizhu Zhou, Na Ta, Qian Qian, and Yuguo Liao

Department of Computer Science and Technology, Tsinghua University, Beijing, China
{fengjh, liguoliang, dcszlj}@tsinghua.edu.cn
{dan04, qqpeter99, liaoyg03}@mails.tsinghua.edu.cn

Abstract. The identification of parent-child or ancestor-descendant relationship between XML elements plays a crucial role in efficient XML query processing. One popular method is to code each node in the XML document tree. However, its main problem is that either lacks the ability to support XML documents update or need huge storage space. This paper proposes a new update-supporting coding scheme based on binary-tree to identify the ancestor-descendant relationship or the parent-child relationship in constant time bound, which also effectively supports XML documents update. To reduce the coding space, we then propose a new storage approach, Blocked Binary-Tree Coding scheme (**BBTC**), whose average code length reduces to $O(\log(n))$. Our extensive experiments show that **BBTC** significantly outperforms previous ones.

1 Introduction

As huge amount of XML (eXtensible Markup Language) data emerge rapidly, the use of XML is not limited to interpret and operate the documents from the Web. How to effectively store and query these XML documents becomes an important issue. A number of query languages have been proposed for querying XML documents, e.g., Lore[1], XML-QL[2], XPath[3], XQuery[4], and so on. One of their core techniques in common is the use of path expressions to express and search user-defined structure modes to implement structure query of XML. Series of XML coding schemes have been brought forward for the query esp. structure query of XML documents [5]. Instead of traversing the whole original document, each node in the document tree is assigned a unique code so that the parent-child or ancestor- descendant relationship of element nodes and attribute nodes in the tree can be worked out directly. Therefore, query of XML can be converted to structure join by coding schemes.

However, there are problems with these recently suggested XML coding schemes: either some of them do not support update of XML documents, which means that it has to recode again whenever XML documents change, or need huge storage. Therefore, in this paper a new update-supporting coding scheme based on binary-tree for XML documents is proposed. It not only efficiently figures out relationship of two elements, but also reduces average code length to $O(\log(n))$, in addition it supports update effectively.

In brief, the major contributions of our **BBTC** work are as follows:

- It can identify the ancestor-descendant or the parent-child relationship in constant time bound, and the identification of such relationships is very simple through equal value operations, i.e. add and shift, rather than non-equal value operations such as estimation of region range etc.
- There is no need of extra space tradeoff, the average code length is $O(\log(n))$, which is asymptotically minimum and state-of-the-art.
- **BBTC** maintains the order information of sibling nodes so that queries on sibling relationship are supported effectively.
- **BBTC** supports XUpdate (<http://xmldb-org.sourceforge.net/xupdate/>) and if XML documents are modified, it does not need to recode the documents again.

This paper is organized as follows: section 2 introduces and analyzes related researches on XML coding techniques; in section 3, our coding scheme is addressed in details; in section 4, a hierarchical storage method based on the binary-tree, **BBTC**, is introduced, whose average code length is $O(\log(n))$; in section 5, experimental analysis of our coding scheme is presented with comparisons to other existing coding schemes; a conclusion of XML coding schemes is addressed in section 6.

2 Related Work

Various kinds of coding schemes have been proposed for query processing of XML documents. Among those, a number of methods that code nodes in XML document trees construct the mainstream. And all the existing coding schemes fall into two main classes: (1) coding based on region and (2) coding based on path. Making use of the order characteristic of XML documents, the first class assigns each node a code according to its document order in the original document; meanwhile, the second class focuses on the nested characteristic of XML documents, allocating a code to each path and each node which can be reached from the root of the tree. At present, coding based on region takes priority.

Region code is one of the most popular among coding schemes. Its main idea is to assign a region code [**start, end**] to each node in the XML document tree, satisfying that a node's region code contains all of its descendants' codes. That is, node (u) is the ancestor of node (v) iff. $start(u) < start(v)$ and $end(v) < end(u)$. Dietz coding scheme was proposed in reference [6]. Li et al proposed the Li-Moon (XISS) coding scheme in reference [7]. Zhang et al proposed the Zhang coding scheme in reference [8]. Wan et al proposed the Wan coding scheme in reference [9]. The ideas in [6], [7], [8] and [9] are basically the same. They estimate relationship between nodes according to their region information. One of their advantages is that they are relatively simple, and the average code length is $O(\log(n))$. But, layer or height information is required when differentiating the ancestor-descendant and parent-child relationships. A disadvantage is that complex operation, instead of equal-value estimation, is used when deciding the relationship between nodes, which means that if there are many nodes, it is not easy to decide relationship among them. Moreover, these methods cannot support XML documents update well. Some researches [9, 10, 11] have proposed a solution which preserves code space for the value of **size** in **<order, size>** or make **order** the extended pre-order traversal number so that extra space can be preserved to support future update

operations. But it is difficult to decide how much space the preservation is to make, and when the preserved space is used up, the XML document has to be recoded again.

Bit-vector coding scheme is proposed in reference [12]. Each node in the XML document tree is assigned an n -bit vector with n being number of the nodes. This method can easily work out the ancestor-descendant relationship but needs $O(n)$ code space. It does not support the update operation either. Dewey code is proposed in reference [13]. It directly uses code of the parent node as the prefix of the child node, which is like catalogue of a book. Since this method needs to employ the prefix when estimating relationship among different nodes, it works slower than the arithmetic operations. Meanwhile, the code space is $O(n)$, which is relatively large. But this method can support the update operation. Wang et al proposed the PBiTree coding scheme in reference [14]. It converts an XML document tree into a binary tree, and numbers each node sequentially. It runs the equal-value operation which is easy for computer implementation. The main difference from our approach is that, it does not support the update operation and involves large storage.

3 A New Update-Supporting Coding Scheme

The coding scheme for XML documents is important for structure query, but current coding schemes have such problems that either they are disadvantageous for updating XML documents or their coding space is too large. This paper proposes a new update-supporting coding scheme which can solve these problems properly.

3.1 Coding Scheme and Algorithm

Our code is in form of a tuple $\langle \text{order}, \text{sibling_order} \rangle$, in which **order** represents the position information of a node in the XML document tree, and **sibling_order** means the sequential number of a node among its sibling nodes. Thus query of sibling relationship is effectively supported. The method to code XML documents is: the **order** of the leftmost child is its parent's **order** multiplying 2, and its **sibling_order** is 0; the **order** of other nodes except for the leftmost child are their left neighboring sibling nodes' **order** multiplying 2 plus 1, and **sibling_order** are their left neighboring sibling nodes' **sibling_order** plus 1 (order of the root is 1, sibling_order is 0).

Algorithm 1: Coding_XML_Tree (N)

{The coding algorithm for an XML document tree T}

Input: Node in T *{N is a node of an XML document tree}.*

Output: Codes of all nodes in T *{T is an XML document tree.}*

begin

if N is Root_of(T) **then** { N.order=1; N.sibling_order=0; }

else

if N is leftmost_child_of(parent(N))

then { N.order=parent(N).order*2; N.sibling_order=0; }

else { N.order=left_neighbor_sibling(N).order*2+1;

 N.sibling_order=left_neighbor_sibling(N).sibling_order+1; }

endif

endif

for each NC, NC is a child of Node(N)

 Coding_XML_Tree (NC);

{recursive calling all the children nodes of N based on depth first search}

endfor

end.

Algorithm 1 lists detailed steps about coding an XML document using our coding scheme, and we can work out codes for all nodes during one time scan of the XML document tree. A simple example (Fig.1.) below gives more details (codes of all TEXT_NODES are omitted).

```

<Bookset>
<Book>
  <ID>1</ID>
  <Author>
    <Name>Fengjh</Name>
    <Degree>PhD</Degree>
  </Author>
  <Name>DB</Name>
</Book>
<Book>
  <ID>2</ID>
  <Author>
    <Name>Zhoulz</Name>
    <Degree>PhD</Degree>
  </Author>
  <Name>DDB</Name>
</Book>
</Bookset>
    
```

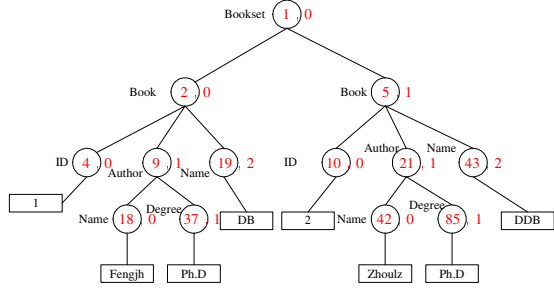


Fig. 1. XML file bookset.xml and corresponding codes

3.2 Properties of the New Coding Scheme

3.2.1 Rules to Infer the Relationship Between Two Elements

We can infer the relationship between any two elements by making use of properties of the new coding scheme through simple operations. The rules for deciding the relationship between two elements are:

Given elements A, D, and N in an XML document tree T.

- (1) $N.layer = \text{number of zeros in the binary representation of } N.order + 1$; this value represents which layer the node N lies in the XML document tree.
- (2) if A is an ancestor of D $\Leftrightarrow 2^* A.order = \left\lfloor \frac{D.order}{2^{\lfloor \log_2 D.order \rfloor - \lfloor \log_2 A.order \rfloor}} \right\rfloor$
 $\Leftrightarrow A.order \ll 1 = D.order \gg (\lfloor \log_2(D.order) \rfloor - \lfloor \log_2(A.order) \rfloor - 1)$
- (3) if A is a parent of D $\Leftrightarrow 2^* A.order = \left\lfloor \frac{D.order}{2^{\lfloor \log_2 D.order \rfloor - \lfloor \log_2 A.order \rfloor - 1}} \right\rfloor$ and $A.layer + 1 = D.layer$
 $\Leftrightarrow A.order \ll 1 = D.order \gg (\lfloor \log_2 D.order \rfloor - \lfloor \log_2 A.order \rfloor - 1)$ and $A.layer + 1 = D.layer$

Note: $\lfloor X \rfloor$ means to get the integer part of number X, \ll means shift leftward, \gg means shift rightward.

3.2.2 Advantages of Our Coding Scheme

(1) Only shift and add operations are needed (refer to section 3.2.1) when inferring the ancestor-descendant or parent-child relationships between two elements. For instance, if we want to identify the relationship between `/Bookset/Book[1]` (order is 2) and `/Bookset/Degree` (1st order is 37, 2nd is 85) in bookset.xml, (see also Fig.1. in section 3.1); meanwhile, $layer(2)=2$, $layer(37)=4$, $layer(85)=4$; $\log_2 2=1$; $\log_2 37=5$; $\log_2 85=6$.

$\therefore 2 * 2 = \left\lfloor \frac{37}{2^{5-1-1}} \right\rfloor$. But, $\text{layer}(2) + 1 \neq \text{layer}(37)$

$\therefore 1^{\text{st}}$ Degree Node is descendant of Book[1] , but not child;

$\therefore 2 * 2 \neq \left\lfloor \frac{85}{2^{6-1-1}} \right\rfloor \therefore 2^{\text{nd}}$ Degree Node is not descendant of Book[1] .

(2) Equal-value inference is employed when deciding the ancestor-descendant relationship instead of non-equal operation, which is advantageous for the structure join operation of massive nodes.

(3) There is no need to attach the layer information of code, since **order** already contains such information (refer to rule 1 in section 3.2.1).

(4) It is advantageous to get the sequential number of a given node among its sibling nodes by **sibling_order**, so that it is easy to get the position of a node in XML document. In addition, **sibling_order** is also in favor of update, we will introduce how to process our codes when update in next section.

3.3 Updating of XML Documents

According to the basic idea, when using the new coding scheme, there is no need to recode the XML document again but only some simple processes based on existing codes when updating the XML document. The following segments about XUpdate's examples are cited from the web site, <http://xmldb-org.sourceforge.net/xupdate/>.

3.3.1 Insert Operation

There are two cases of the insert operation. The first case is to insert a node as sibling node of a certain node with an appointed position. e.g.:

```
<xupdate:modifications version="1.0" xmlns:xupdate="bookset.xml">
  <xupdate:insert-before select="/Bookset/Book[1]/Author/Degree">
    <xupdate:element name="Sex">male</xupdate:element>
  </xupdate:insert-before>
</xupdate:modifications>
```

The above example means: to insert *Sex* information to the first *Author* of the first *Book* and as *Degree*'s left sibling , simply find the given referencing node $\langle \text{Degree} \rangle$ (represented as **N**) whose code is $\langle \text{N.order}, \text{N.sibling_order} \rangle$ and its sibling node **NR** with the maximal **order** code, then insert the new node with code as $(\text{NR.order} * 2 + 1, \text{N.sibling_order})$. At last, increase **sibling_order** of all sibling nodes of **N** (including **N** itself) whose **sibling_order**'s values are not less than **N.sibling_order** by one. In the given example, we only need to insert a new node $\langle \text{Sex} \rangle$ whose code is (75, 1), with the code of $\langle \text{Degree} \rangle$ changing into (37, 2).

The second case is to add a node as a child node of a certain node but not appointing sequence among its siblings. e.g.:

```
<xupdate:modifications version="1.0" xmlns:xupdate="bookset.xml">
  <xupdate:append select="/Bookset/Book[1]/">
    <xupdate:element name="Author">
      <Name>Zhoulz</Name>
    </xupdate:element>
  </xupdate:append>
</xupdate:modifications>
```

The above example means: to insert an *Author* node as the child node of the first *Book*. We find the code `<N.order, N.sibling_order>` of the given insertion position of the referencing node `<Book[1]>` (represented as **N**), then find its child node **NR** with the maximal **order** code and last child node **NS** with the maximal **sibling_order** code, then insert the new node accordingly. The code of the new node can be computed by $(NR.order * 2 + 1, NS.sibling_order + 1)$. In the above example, `<Author>` node $(19 * 2 + 1, 3)$ and its child node `<Name>` $(39 * 2, 0)$ need to be inserted. Algorithm about inserting a new node into an XML document tree is omitted due to the space limited.

3.3.2 Delete Operation

```
<xupdate:modifications version="1.0" xmlns:xupdate="bookset.xml">
<xupdate:remove select="/Bookset/Book[2]/Author/Name"/>
</xupdate:modifications>
```

This example tries to delete the `<Name>` node of the *Author* of the second *Book*. The code of this node needs to be removed, and decrease the **sibling_order** of `<Name>`'s sibling nodes whose **sibling_order** is greater than **Name.sibling_order** by one.

3.3.3 Update Operation

```
<xupdate:modifications version="1.0" xmlns:xupdate="bookset.xml">
<xupdate:update select="/Bookset/Book[2]/Author/Name">Fengjh</xupdate:update>
</xupdate:modifications>
```

This example tends to change the *Author* name of the second *Book*. There is no need to change the code but simply change this node's value.

The above three sections present the new update-supporting coding scheme and its properties. How to infer the ancestor-descendant relationship also has been covered. In addition, we point out the advantages of this coding scheme and present some necessary but simple processes needed to be carried out when updating XML documents. The only disadvantage of the scheme is that it needs relatively large coding space. Hence, we will explain how to solve the problems.

4 Storage of the Binary-Tree Code

General XML document trees are not regular, but the binary tree has such advantages that they are hierarchical and easy to store, therefore general XML document trees need to be converted to ordered binary trees in order to use these advantages.

4.1 Conversion from XML Document Tree to Binary-Tree

Let T be an XML document tree and BT be the correspondent converted binary tree, the corresponding conversion rules are as follows:

- (1) $A \in T$, if $A = \text{root_of}(T)$ then $A = \text{root_of}(BT)$
- (2) $\forall A, D \in T$, if $D = \text{first_child_of}(A)$ then $D = \text{left_child_of}(A)$ in BT
- (3) $\forall D_1, D_2 \in T$, if $D_2 = \text{right_sibling_of}(D_1)$ then $D_2 = \text{right_child_of}(D_1)$ in BT

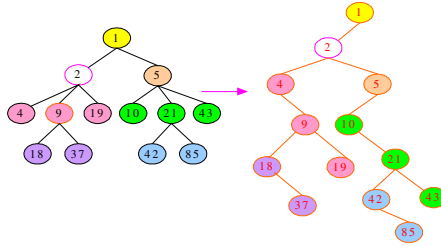


Fig. 2. XML document tree of bookset.xml and corresponding converted binary tree According to the above rules, we can convert the XML document tree in Fig.1. into the binary-tree in Fig.2

4.2 Coding Method of Binary-Tree

The binary-tree coding method makes use of order characteristics of the XML document tree, and it codes nodes according to their positions in the complete binary tree. The detailed coding method is as follows:

- (1) root_of(BT).order = 1
- (2) $\forall D1, D2, A \in BT$
 - if D1 is left_child_of(A) then $D1.order = A.order * 2$
 - if D2 is right_child_of(A) then $D2.order = A.order * 2 + 1$

The coding method of the binary-tree is identical with Algorithm 1 in section 3.1, we can obviously draw the conclusion from Fig. 2.

4.3 Storage Strategy

Since the only disadvantage of the binary-tree code is that it needs relatively large coding space, we have to solve this issue. There are two feasible ways to solve it:

Compression: Due to properties of binary-tree, the adjacent codes have strong similarity in that they have the same ancestor nodes. Therefore, many nodes have the same prefix, i.e., there is great redundancy in the codes. In other words, much work can be done using data compression.

Hierarchical Storage: Due to properties of binary-tree, some offspring nodes store information of their ancestors repeatedly. Therefore, we could employ hierarchical storage method to process these codes. In other words, we could partition the Binary-tree into different sub-blocks with nodes in each sub-block have the same ancestor and code each node relative to the ancestor (root of the sub-block) in each sub-block. In other words, common information of a sub-block is stored in its header so that the storage space can be reduced.

Obviously, compression of codes will reduce the storage space greatly. However compression itself and decompression will undoubtedly require extra time spending when inserting a new node. Performance of query processing is affected accordingly. Therefore, we use the hierarchical method to store codes.

4.4 BBTC

Problems must be settled when partitioning the binary-tree are, how big a sub-block should be and what structural relationship should be maintained between sub-blocks. The scale of a sub-block directly affects performance of storage. If a sub-block is too small, it is not good for nodes aggregation; while if it is too big, there is too much redundancy in each sub-block. Before we introduce hierarchical storage in detail, present these following definitions:

Definition 1. Non-trivial leaf node

If node N is a leaf node of a sub-block but not a leaf node of the whole binary tree, it is called a **non-trivial leaf node**.

Definition 2. Non-trivial root node

If node N is the root node of a sub-block but not the root of the whole binary tree, it is called a **non-trivial root node**.

Definition 3. Inner node

All nodes in each sub-block, are called this sub-block's **inner nodes**.

4.4.1 Division of the Storage Structure and Sub-block

Due to properties of the binary-tree, some descendent nodes store information of their ancestors repeatedly. Therefore, we could employ hierarchical storage method to process these codes. In other words, we could partition the binary-tree into different sub-blocks with nodes in each sub-block having the same ancestor, and then code each node relative to the ancestor in each sub-block. Therefore, common information of a sub-block is stored in its header so that the storage space can be reduced, and we call it Blocked Binary-Tree Coding scheme (**BBTC**). The common prefix called BlockID (**BID**), and the other part of the code to distinguish each other in the sub-block is called InnerID (**IID**). And let B be the total number of bits needed to code one node in a sub-block, which is also the height of a sub-block.

The header information of a whole block is not only helpful for inferring ancestor-descendant relationship but also avoids searching between sub-blocks. Steps for partitioning sub-blocks are:

(1) All nodes whose height differences to the root are less than B including the root itself are in a same sub-block. The first sub-block's **BID** is 1 and **IID** is 1, whose root is the root of the binary tree.

(2) If some node N_D is not allocated to any sub-block, and its parent node N_A is a non-trivial leaf node of some sub-block, then create a new sub-block with N_D as its non-trivial root, and allocate its descendant nodes whose height difference to N_D are less than B into this sub-block. Suppose that the **IID** of N_A is C , and the **BID** of this sub-block which N_A belongs to is D , then the **BID** of the new block rooted at N_D can be computed by **formula 1**:

$$BlockID = \begin{cases} ((D - 1) * 2^{B-1} + C) * 2 & \text{if } N_D \text{ is left-child of } N_A \\ ((D - 1) * 2^{B-1} + C) * 2 + 1 & \text{if } N_D \text{ is right-child of } N_A \end{cases} \quad (1)$$

(3) The **IID** of each sub-block's root is always 1, and then code other inner nodes in each sub-block using the method in section 3.1 or 4.2.

- (2) If the sub-blocks that these two nodes belong to are **sibling sub-blocks** or **collateral sub-blocks**, then there is no ancestor-descendant relationship between the two nodes.
- (3) Otherwise, suppose **IIDs** are C1 and C2 respectively and their **BIDs** are D1 and D2, then we infer their relationship by inferring relationship between N1 and D2 through rules in section 3.2.1. Here, $N1 = (D1 - 1) * 2^{\lceil \log_2 C1 \rceil} + C1$ (suppose $D1 < D2$). While the layer number of a node is $C.layer + D.layer - 1$ (C and D are respectively its **IID** and **BID**).

Through these three rules, we can correctly and completely infer the relationship between any two nodes. For instance, we infer the relationship between node 3(9) in sub-block 9 and node 1(85) in sub-block 85 (such as, node 3(9) denotes **BID**=9 and **IID**=3, and so on). Since 9 and 85 have no ancestor-descendant relationship, therefore, sub-block 9 and sub-block 85 are collateral sub-blocks, and the two nodes have no ancestor-descendant relationship. Then we infer the relationship between node 3(10) in sub-block 10 and node 1(85) in sub-block 85. Since 10 is the ancestor of 85, so judging relationship between $(10-1)*2+3=21$ and 85. We conclude that these two nodes are ancestor and descendant. And their layers are $3(layer(3)+layer(10)-1=3)$ and $4(layer(1)+layer(85))-1=1+4-1=4$, so they are also parent-child relationship.

4.5 Processing of the Insert Operation

When updating XML documents, the update and delete operations can be carried out following operations in section 3.3, however, the insert operation needs some extra cost. If the parent node of the insert node is a non-trivial leaf node of a sub-block, we construct a new sub-block taking it as the root of this sub-block and calculate its **BID** by formula 1, and its **IID**=1, then modify **sibling_order** according to the section 3.3. Or otherwise, the new node should be inserted into the sub-block in which its parent node is according to the section 3.3.

5 Experimental Analysis of the Binary-Tree Code

Currently, there are mainly Region Code, Bit-Vector Code, Prefix Code and PBiTree for coding XML documents, however Region Code is the most popular presently, therefore we compare BBTC with it. **BBTC** rapidly infers the ancestor-descendant relationship, reduces the average code length to $O(\log(n))$ and supports update efficiently. Therefore we devised several groups of experiments using the standard XMark and Shakespeare data to test our approach in space performance and time performance. XMark was generated from standard data [15] and Shakespeare used standard Shakespeare 2.00 data [16]. The experimental environment is: windows 2000 server, AMD2600 CPU, 1G RAM. We used standard C++ for programming.

5.1 Experiment 1: To Determine Value of B in BBTC

The **BBTC** will reduce storage space evidently, thus the space cost is less than the Region Code, and we have to determine the value of B firstly. In Fig.4. (a) and (b)

illustrate the choice of B for Shakespeare and XMark data sets respectively. They present the space of the **BBTC** consumed with different values of B for various XML data sets. Due to the lowest point of each curve corresponding to the minimum space cost, the value of B at this point is its best choice. On the one hand, the curves with block-partitioning have observable advantages in storage size, but the stability of space cost with different values of B is another advantage revealed.

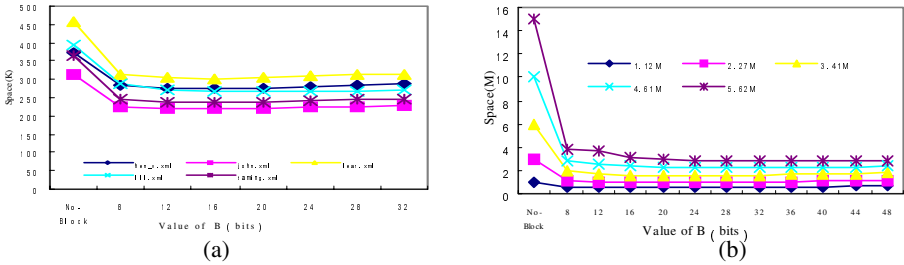


Fig. 4. Fix on B in XML doc

In Fig.5. (a) and (b) illustrate the effect of different values of B on space performance for Shakespeare and XMark data sets respectively. They present the space of the **BBTC** consumed with different XML documents for various values of B. Obviously, the cost of space is evidently reduced with **BBTC**. Compare with Non-block, **BBTC** reduces 30% in Shakespeare data, but it reduces 73% in XMark data.

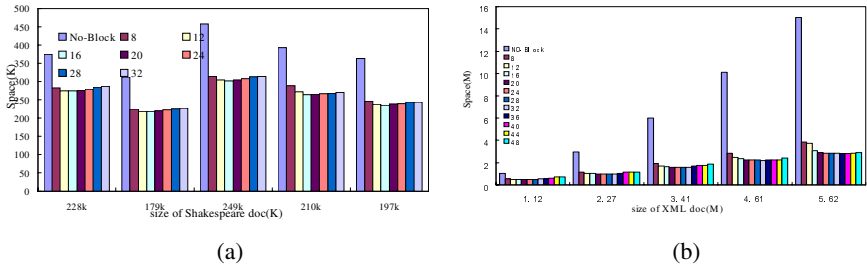


Fig. 5. the effect of different B on space performance

We concluded from our analysis that the best B value has to do with the height of the binary tree converted from the XML document tree. And it is also related to the number of nodes in the XML document. Fig.6. indicates the relation between the best B and the number of nodes in the XML document. Due to the relation between the best B and the number of nodes meets logarithmic normal distribution, we make a conclusion that the best B for one XML document can be represented by $\log(n)$ approximately, in which n is the number of nodes in the XML document.

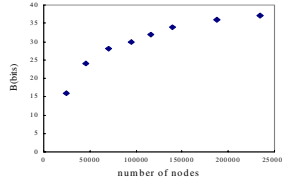


Fig. 6. Relation between the best B value and the number of nodes in XML doc

5.2 Experiment 2: Comparison of Several Coding Algorithms

BBTC has superior storage performance to the Region Code, this is because the Region Code maintains two numbers for one region and its performance decreases. The Region Code is not as good as the BBTC in time performance either, since it has to scan the XML documents at least twice.

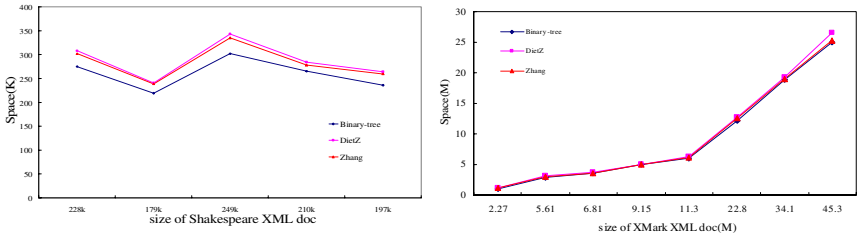


Fig. 7. Space performance comparison of several coding schemes

We compared the BBTC to Dietz and Zhang Code experimentally. The data sets are also from the XMark and Shakespeare. Fig.7. & Fig.8. show BBTC better than other coding schemes in space and time respectively. Compare with Region Code, BBTC reduces 16% in space and 25% in time with Shakespeare data, and reduces 8% in space and 40% in time with XMark data when its size of the data reaches 45.3M.

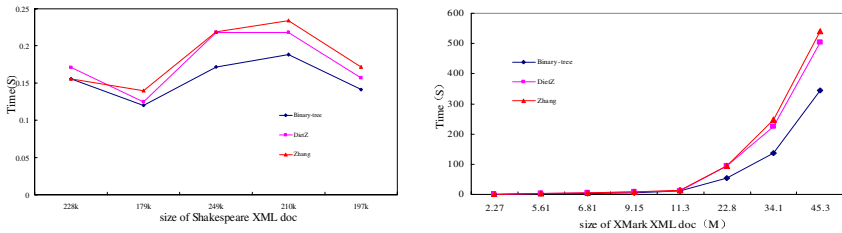


Fig. 8. Time performance comparison of several coding schemes

6 Conclusion

In this paper, we propose the new update-supporting coding scheme based on binary-tree, which not only codes the XML document and infers relationship between

nodes rapidly, but also solves the problem of those previous schemes that documents have to be recoded again when update. A hierarchical storage method, **BBTC**, is also presented, which can reduce the average code length to $O(\log(n))$. Experiments have proved that BBTC has relatively strong query processing ability than naïve ones.

References

1. S Abiteboul, D Quass, J McHugh et al. The Lore query language for semi-structured data I Int'l Journal on Digital Libraries , 1997 ,1 (1) :68-88
2. Alin Deutsch , Mary Fernandez , Daniela Florescu et al. A query language for XML. The 8th Int'l World Wide Web Conf , Toronto , 1999
3. Jamex Clark , Steve DeRose, XML path language (XPath), W3C Recommendation I World Wide Web Consortium , 1999
4. Don Chamberlin , Daniela Florescu , Jonathan Robie et al. XQuery : A query language for XML. W3C working draft, WWW, 2001
5. D. Florescu, D. Kossman et al. Storing and Querying XML Data using an RDBMS. IEEE Data Engineering Bulletin, Vol. 22, No. 3, September 1999
6. Paul F Dietz. Maintaining order in a linked list. The 14th Annual ACM on Theory of Computing , San Francisco , 1982
7. Quanzhong Li and Bongki Moon. Indexing and querying XML data for regular path expressions. VLDB 2001
8. Zhang C, Naughton J, DeWitt D et al. On Supporting Containment Queries in Relational Database Management Systems. SIGMOD, California, May 2001. 426 -437
9. Dao Dinh Kha, Masatoshi Yoshikawa, and Shansake aemara. An XML indexing structure with relative region coordinate. ICDE 2001
10. Lu Yan , Zhang Liang , Wang Wei and Shi BaiLe. A New XML Document Coding Scheme,. Journal Of Computer Research And development, Vol.141, No.13, March 2004
11. Luo Daofeng, Meng Xiaofeng. Updating of Extended Preorder Numbering Scheme on XML, Computer Science Vol.30, No. 10, October 2003
12. N.Wirth. Type Extentions. Acn Transaction on Programming Languages and systems 1988, 10(2):204~214
13. Igor Tatarinod, Stratis D, Kedin Beyer et al. Storing and querying ordered XML using a relational database system. SIGMOD 2002
14. Wei Wang, Haifeng Jiang, Hongjun Lu, and Jeffrey Xu Yu. PBiTree coding and efficient processing of containment joins. ICDE 2003
15. <http://www.xml-benchmark.org/>
16. <http://www.xml.com/pub/r/396>

Using XML Structure to Reduce Candidate Nodes Participated in Query Processing¹

Zhenying He, Jianzhong Li, Chaokun Wang, Pengcheng Ge, and Haikun Chen

Department of Computer Science and Engineering, Harbin Institute of Technology, China
{hzy, lijzh, chaokun, grey_hit}@hit.edu.cn
chark1983@hotmail.com

Abstract. Several algorithms have focused on processing path expression queries. Following those algorithms, all the nodes, matched with path expressions, are participated in computing. In this paper, we propose a novel filter strategy to reduce the number of candidate nodes based on the structure of XML data. All nodes are clustered based on their labels, and path information of each node is kept in bit vectors. Our filter technology mainly depends on high performance of bit operations. The experimental results show that these filter algorithms are effective, scalable and efficient.

1 Introduction

A number of algorithms have been proposed to answer and accelerate path expression queries. Those algorithms, however, only make use of parent-son and ancestor-descendant relations between element nodes when query processing. All the nodes, matched with path expressions, are participated in computing. Accordingly, CPU and I/O costs are wasted in dealing with these useless nodes.

Some works make use of range-based coding method to build a special index, namely *XR-Tree*[1], and only considered skipping by the coverage relationship between two nodes. As a result, the skip operations must follow the path steps one by one. However, the skip will be disabled in some cases. For example, the query expression, against the data whose schema graph is shown in Fig.1(a): $Q_2=//a/e$ matches e elements that have a element as their father. Following the methods of the structural join[2-7] or the twig join[8-10], all the e -nodes will participate the query processing and none of e -nodes will be skipped. In fact, the set of e -nodes is consisted of $\{e_b\}$, $\{e_f\}$ and $\{e_e\}$, where $\{e_b\}$ is the set of e -nodes passing b nodes, $\{e_f\}$ is the set of e -nodes via f -nodes and $\{e_e\}$ is the set of e -nodes who have a parent named a . Thus only the elements in $(\{e\} - (\{e_b\} \cup \{e_f\}))$ will help bring about the results. It is efficient to skip the e -nodes in $(\{e_b\} \cup \{e_f\})$ and compute the result against $(\{e\} - (\{e_b\} \cup \{e_f\}))$.

In this paper, we propose a novel filter strategy to reduce the number of candidate nodes. All nodes are clustered based on their labels, and path information of each node is kept in bit vectors. We first calculate the expression of filter operation according to the query and the structure of data (Sec. 3). Then, we apply several methods to

¹ Supported by the Defence Pre-Research Project of the “Tenth Five-Year-Plan” of China No.41315.2.3; the National Natural Science Foundation of China, No. 60273082.

skip more nodes (Sec. 4). To the best of our knowledge, no previous work has addressed to reduce the number of candidate nodes based on XML schema information. The contributions of this paper can be summarized as follows:

1. We propose a novel filter strategy to skip the elements in useless branches. Following this strategy, filter expression is calculated by exploring the XML schema graph. We also provide pruning strategies to generate the filter expression efficiently.
2. We develop several filter algorithms against indexed XML data. Our technology mainly depends on high performance of bit operations. For the sparseness of bit vectors, we also develop compressed-based and signature-based algorithms to handle the filter strategy. The experimental results demonstrate that our techniques have good performance and better efficiency.

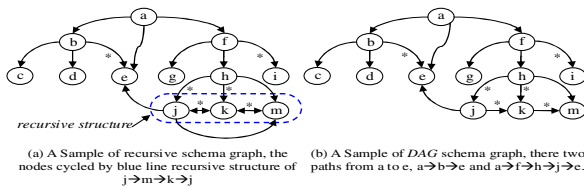


Fig. 1. Two sample XML schema graphs

The rest of this paper is organized as follows. Section 2 gives some preliminary knowledge on XML schema graph. The filter strategy and the approach for computing the filter expression are discussed in Section 3. Section 4 addresses the filter algorithms based on bit vectors. And section 5 presents the performance study. Finally, we conclude this paper in Section 6.

2 Preliminaries

An XML schema can be viewed as a directed graph $SG = (V, E)$, where V is the set of vertices and E is the set of edges. The vertices correspond to elements and attributes and the edges represent parent-child relationships. A sample non-recursive schema graph is given in Fig.1(b). If the schema graph is a tree, then we call it a *Tree-structure* schema graph. If it is acyclic, we call it a *DAG-structure* schema graph (directed acyclic graph). Otherwise, it is a *recursive-structure* schema graph. The instance of an XML schema is a tree, which is called as data tree. We also give an example data tree in Fig.2, which confirms to the XML schema graph illustrated in Fig.1(a).

Definition 1. Let $SG=(V,E)$ is an XML schema graph. There exists father-son relation from n_1 to n_2 , denoted by $n_1 \rightarrow n_2$ or $n_2 \leftarrow n_1$, if $(n_1, n_2) \in E$. There exists ancestor-descendant relation from n to m , denoted by $n > m$, if: (i) $n \rightarrow m$ or (ii) there exists k , satisfying $n > k$ and $k > m$.

In the example of Fig.1(a), there exists father-son relation between a and b . Note that there exists father-son relation between j and k , that is, $j \rightarrow k$ and $k \rightarrow j$.

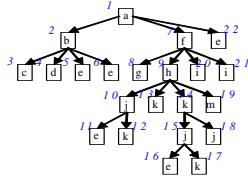


Fig. 2. Data tree for Fig. 1(a)

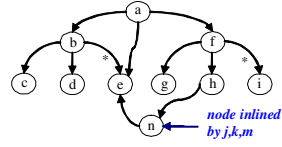


Fig. 3. ISG for Fig. 1(a)

Definition 2. Let $\alpha = \{n_1, n_2, \dots, n_m\}$ is a finite set of nodes. S is the *isolated node* in α , if $\forall R \in \alpha (R \neq S)$, not satisfying $R \rightarrow S$ and $S \rightarrow R$.

Definition 3. Given an XML schema graph $SG = (V, E)$. SG is *recursive-structured*, iff there exists $n \in V$, satisfying $n > n$. SG is *non-recursive-structured*, if it is not recursive-structured.

Definition 4. Let $SG = (V, E)$ is a non-recursive-structured XML schema graph. SG is *tree-structured*, for any $n_1 > n_2$, if $n_1 > n_3 \wedge n_3 > n_2 \wedge n_1 > n_4 \wedge n_4 > n_2$, satisfying $n_3 = n_4$ or $n_3 > n_4 \wedge n_4 > n_3$. SG is *DAG-structured*, if SG is not tree-structured.

3 Filter Strategy

The start of this section is primarily a presentation of precisely defined notations. Its purpose is more to ensure your use of notation reduces the number of candidate nodes than only to acquire mathematical concepts. Then, subsection 3.2 delves further into the method of generating the filter expression. For the limitation of space, the proof of lemmas and theorems can not be included in this paper.

3.1 Filter Strategy

In an XML schema graph SG , it is *reachable* from e_1 to e_2 , denoted by $e_1 \Rightarrow e_2$, if $e_1 > e_2$ or $e_1 = e_2$. It is *reachable* between e_1 and e_2 , if $e_1 \Rightarrow e_2$ and $e_2 \Rightarrow e_1$, denoted by $e_1 \Leftrightarrow e_2$.

Lemma 1. Let $SG = (V, E)$ is an XML schema graph. The binary relation \Leftrightarrow on V is a relation of equivalence.

Example 1. Suppose ISG is the example schema graph given in Fig. 1(a). It is reachable from node a to every node in ISG, not reachable from node f to node b , reachable between j , k and m . Thus, j , k and m must all lie in the same block of \Leftrightarrow .

Definition 5. Given an XML schema graph $SG = (V, E)$. $ISG = (V_{ISG}, E_{ISG})$ is called as the *inlined schema graph* of SG , and is defined as follows: (1). there exists a function $F_V: V \rightarrow V_{ISG}$, $\forall n_1, n_2 \in V$, satisfying $n_1 \Leftrightarrow n_2$, iff $F_V(n_1) = F_V(n_2)$; (2). there exists a function $F_E: E \rightarrow E_{ISG}$, satisfying $\forall (n_1, n_2) \in E$, iff $(F_V(n_1), F_V(n_2)) \in E_{ISG}$.

The nodes in the same block in V is partitioned into one node determined by \Leftrightarrow . Fig. 3 give the inlined schema graph ISG of schema graph shown in Fig. 1(a). In ISG , node j , k and m are inlined as a new node n .

Lemma 2. Suppose that $SG=(V,E)$ is an XML schema graph, and $ISG=(V_{ISG}, E_{ISG})$ is its inlined schema graph. SG is a non-recursive-structured schema graph if $SG=ISG$.

Definition 6. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $a, b \in V_{ISG}$, and $a \Rightarrow b$. A reachable path from a to b is a sequence of $a, n_1, n_2, \dots, n_{k-1}, n_k, b$, where $a > n_1, n_1 > n_2, \dots, n_{k-1} > n_k, n_k > b$.

Definition 7. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $T=(V, E, root)$ is a data tree of ISG , and $p=a, n_1, n_2, \dots, n_{k-1}, n_k, b$ is a reachable path from a to b . The p -reachable set in T , denoted by $\{p(a, b)\}$, is consists of the b nodes via the path p .

Lemma 3. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $T=(V, E, root)$ is a data tree of ISG , and $\{node(e_i)\}$ is the set of all nodes labeled e_i . $\forall x \in \{node(e_i)\}$, there exists only one reachable path from $root$ to e_i .

Lemma 4. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $T=(V, E, root)$ is a data tree of ISG . If there exists only reachable path p_i from $root_{ISG}$ to e , then $\{node(e)\}=\{p_i(e)\}$.

Theorem 1. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $T=(V, E, root)$ is a data tree of ISG , $\{node(e)\}$ is all the nodes of e in V_T , p_1, \dots, p_n is all the reachable path from $root_T$ to e . $\{node(e)\}=\{p_1(e)\} \cup \{p_2(e)\} \cup \dots \cup \{p_n(e)\}$.

Theorem 2. Let $ISG=(V_{ISG}, E_{ISG})$ is an inlined schema graph, $T=(V, E, root)$ is a data tree of ISG , $\{node(e)\}$ is all the nodes of e in V_T , p_1, \dots, p_n is all the reachable path from $root_T$ to e . $\{p_i(e)\}=\{node(e)\} - (\{p_1\} \cup \dots \cup \{p_{i-1}\} \cup \{p_{i+1}\} \cup \dots \cup \{p_n\})$.

3.2 Algorithm for Generating the Filter Expression

The method to computing filter expression is presented in algorithm 1. Two aided data structures, stack chain and hash table chain, are applied to store the intermediate data. The length of these two structures is the number of the stage. The purpose of stack chain is to store the scanned nodes in that stage, while the hash table chain, to store the definitely visited nodes in that stage. This algorithm falls into 3 stages: (i). the stage for initializing from line 1 to line 4; (ii). searching the state space tree in line 5, and (iii). stitching the intermediate result in hash table chain and computing the final filter expression in line 6.

Algorithm 1 *FindFE(p, IG)*

Input: p is the path expression; IG is the ISG ;

Output: the filter expression FE ;

begin

```

1: InitEx(p);
2: stage=0;
3: for each vertex  $u \in V[IG]$ 
4:   do color[u] ← WHITE;
5:   SCAN(root, root, stage);
6:   return MergeHT();

```

end.

For the sample query of $Q=//h/c/e$, it is decomposed to 3 stages: ($root, h, AD$), (h, c, PS) and (c, e, PS) in the stage of InitEx. Meanwhile, a stack and a hash table are

initialized for each stage. Furthermore, all the nodes in inlined schema graph are marked as WHITE (unvisited).

In the procedure of *SCAN*, pruning strategy, shown in line 6-line 8, is applied to reduce the state space tree, while the forward skipping strategy, shown in line 11-line 12, and in line 15-line 20, is applied to determine the location to store those intermediate results. Following the pruning strategy, the forward node will be stored into the current position of hash table chain if that node has already been visited, and the relation of current stage is the relation of ancestor-descendant. Following the forward skipping strategy, if the forward node emerges in other stages, the current stage will be changed accordingly so that the intermediate result will be stored in proper structures. After the computing, the definitely passed nodes of each stage are stored in corresponding hash table, which will be merged together by applying the procedure *MergeHT*.

```

procedure SCAN(u, s, stage)
Input:    u is the start node of this stage;
           s is the current scanned node;
           stage is the number of stage;

begin
1:  if stage ≥ MAX_STAGE
2:    return;
3:  color[u] ← GRAY;
4:  Stack[stage].PUSH(u)
5:  for each v ∈ Adj[u] do
6:    if color[v] = GRAY
7:      if ((IsExisted(HT[stage], v) && (Rel[stage] = AD)))
8:        Add(HT[stage], u);
9:    else
10:     x ← stagenode(v);
11:     if x > stage
12:       SCAN(v, v, x);
13:     if x < stage
14:       SCAN(v, s, stage);
15:     if x = stage
16:       if (Rel[stage] = AD)
17:         Stack[stage] → HT[stage];
18:       else //FS
19:         if (Depth(s, v) = 1)
20:           Stack[stage] → HT[stage];
21: Stack[stage].POP();
end.

```

The procedure of *MergeHT* is used to stitch the intermediate results and require the final filter expression. *MergeHT* is started with computing the complement set RH_i of every hash table in line 1-line 2. Then in line 3, the set of these filtered nodes are required by calculating the intersection of all the RH_i . At last, return the filter expression in line 4.

```

procedure MergeHT();
Output:  the filter expression FE;
begin
1:  for each  $HT_i$  of Hash Table Chain

```

```

2:   do Calculate complement of  $HT_i$ ;
3:   ListC ← Intersection of each  $HT_i$ ;
4:   return elements in ListP and  $\neg$ elements in ListC;
end.

```

It is obvious that the space cost of this algorithm is $9n+n^2$. As to the running time, the cost of scanning in depth-first order is reduced to $O(n^2)$ because each edge in inlined graph is visited only once. While the cost for other parts of this algorithm is $O(n)$. Therefore, the total amount of work is now bounded by $O(n^2)$.

4 Filter Algorithms

In this section, three filter algorithms are presented: BFX-Filter, SX-Filter and CX-Filter.

The structure of bit vectors for node b is shown in Fig. 4. The *BFX-Filter* algorithm works as follows. It scans the bit vectors with the help of a *Window* which size is generally equal to m . It first aligns the up ends of the window and the bit vectors, then execute the filter expression on the slices of these bit vectors in that window, and after that, it shifts the window downwards. It repeats the same procedure again until the bottom end of the window goes beyond the end of bit vectors.

Extended Vector	011111000000000000000000
Solid Vector	010000000000000000000000

Fig. 4. Bit vector for node b

The following pseudocode is the basic *BFX-Filter* algorithm. The input parameters are path expression p and inlined schema graph IG . The *BFX-Filter* algorithm starts with the algorithm of *FindFE* in line 1. From line 2 to line 7, it loads the bit vectors into memory. In view of the object node in path expression, line 4 determines which vector should be loaded. At last, execute the bit operation according to the filter expression from line 8 to line 10.

Algorithm *BFX-Filter* (p, IG)

Input p is the path expression;
 IG is the Inlined Schema Graph;

Output List L of Candidate nodes;

begin

```

1:   $fe = \text{FindFE}(p, IG)$ ;
2:   $enode = \text{GetLastNode}(p)$ ;
3:  the extended bit vectors of nodes in  $fe$ , except
      $enode \rightarrow node\text{-list}$ ;
4:  if ( $\text{IsExtended}(enode)$ )
5:     Append Extended bit vector of  $enode$  to  $node\text{-list}$ ;
6:  else
7:     Append Solid bit vector of  $enode$  to  $node\text{-list}$ ;
8:  for each 32 rows of loaded bit vectors do
9:     if ( $\text{GetFilter}(fe)$ )
10:      Add  $position$  to the tail of  $L$ ;
end.

```

In general, the bit vectors can be organized as a number of integers. Accordingly, the window size of m is set as 32. To further improve the performance of *BFX-Filter* algorithm, we store the slices of the bit vectors in the cache of processor directly, not in memory, when they are loaded. The cost of the running time is $O(n)$.

The *BFX-Filter* algorithm scans the bit vectors in up-down order, regardless whether the vector is sparse or not. As a result, unnecessary bitwise operations are executed during shifts. To handle this problem, we propose the signature-based *SX-Filter* algorithm. A signature file uses a signature that maps several bits to bit masks of 1 bit. It divides the bit vector in blocks of b bit each. To each vector block of size b , a bit mask of size 1 will be assigned. This mask is obtained by checking the existence of 1 bit in the vector block. Here, we use two signature files to store the sequence of bit masks of all blocks. For instance, the sample signature files for the given bit vector with the size of 30 are illustrated in Fig. 5. In that example, the size of block is 3. The bit vector of 0-V is the sequence of masks of bit 0, while the 1-V, bit 1.

1-V:	1	0	1	1	0	1	1	0	1	1	0	0	0													
0-V:	1	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0-V:	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

Fig. 5. Two signature files for the given original vector

Different from *BFX-Filter*, the *SX-Filter* algorithm shifts the window and checks the mask of this block. Whenever ($B_i=1$), therefore the vector block may contain the needed bit. Hence, the filter algorithm must be performed to verify the exact positions in that block. This algorithm is more efficient to filter nodes. This is because a number of blocks are skipped and loaded into the cache in processor.

Algorithm *SX-Filter* (p , IG)

Input p is the path expression;
 IG is the Inlined Schema Graph;

Output List L of Candidate nodes;

begin

```

1:  $fe = \text{FindFE}(p, IG)$ ;
2:  $enode = \text{GetLastNode}(p)$ ;
3: Load the bit vectors of nodes, except  $enode$ , in  $fe$ ;
4: if ( $\text{IsExtended}(enode)$ )
5:   Load Extended Signature Vector of  $enode \rightarrow SV$ ;
6: else
7:   Load Solid Signature Vector of  $enode \rightarrow SV$ ;
8: while ( $\text{!IsEOF}(SV)$ ) do
9:   if ( $\text{GetFilter}(fe)$ )
10:    Add  $position$  to the tail of  $L$ ;
```

end.

Some additional storage costs are introduced in the methods above. This is because the sparseness of the bit vectors. Hence, we also develop a compression-based algorithm, named as *CX-Filter*, to skip nodes. The algorithm of *CX-Filter* makes use of header compression method [11]. This algorithm operates directly on compressed vectors without the need to first decompress them, and therefore, are efficient for sparse bit vectors. It applies *header* that is vector of counts such as $(u_0, c_0, u_1, c_1, \dots, u_i, c_i, \dots, u_s, c_s)$, in which odd-positioned numbers are counts of 1 from the header of bit

vector, and even-positioned numbers are counts of 0 from the header. We give an example in Fig. 6, where *LB* denotes the original bit vector and *H* is its compressed header file. The algorithm for building the header file is more to scan the data or *LB*, then calculate the *Header*. The cost of running time is $O(n)$.

```

LB: 111100000001110000001100111111110000
H:   4       7 7       13 9 15       18 19
    
```

Fig. 6. An example for *header compression*

```

Algorithm CX-Filter (p, IG)
Input      p is the path expression;
             IG is the Inlined Schema Graph;
Output    List L of Candidate nodes;
begin
1: fe=FindFE(p, IG);
2: enode=GetLastNode(p);
3: Load the bit vectors of nodes, except enode, in fe;
4: if (IsExtended(enode))
5:   Load Extended Header of enode;
6: else
7:   Load Solid Header of enode;
8: while (!IsEOF(Header)) do
9:   GetthePosition(Headers) → position;
10:  if(GetFilter(fe))
11:    Add position to the tail of L;
end.
    
```

The *CX-Filter* algorithm shifts the window on Header files and calculates the longest shift in header files (line 8-line 11). The cost of running time is $O(n)$.

5 Experimental Results

We implemented these algorithms in C++, and carry out our experiments on a Windows XP machine with Celeron 1.7GHz CPU and 256M main memory. For our experiments, we use the benchmark of XMark, whose factor is ranged from 1 to 4. The test queries that we used are shown in Table 1. We chose these queries for the following reasons. Q1 evaluates recursive (*//*) XPath queries. Q2 and Q3 are similar to Q1 but do not use recursion. The last query was chosen to test the number of elements scanned. And we evaluated the performance of these filter algorithms using the following metrics: (1) *disk storage requirements*; (2) *the number of elements scanned*; (3) *running time*; (4) *the number of pages access*.

Table 1. Test Queries

<i>Query name</i>	<i>Query</i>
Q1	<i>//regions//item/name</i>
Q2	<i>//regions/Asia/item</i>
Q3	<i>//category/description</i>
Q4	<i>//person/address</i>

5.1 Disk Storage Requirements

This metric reflects the external disk space requirements for building indices. And it is measured by the ratio of the index size to the original data size. Note that, in Fig. 7, the space cost of CX-Filter is lower than other two filter methods, no matter what is the value of XMark factor.

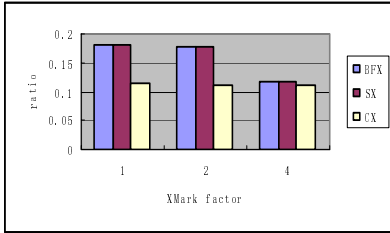


Fig. 7. Storage requirements

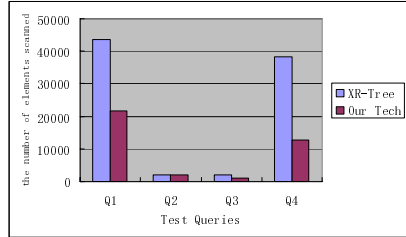
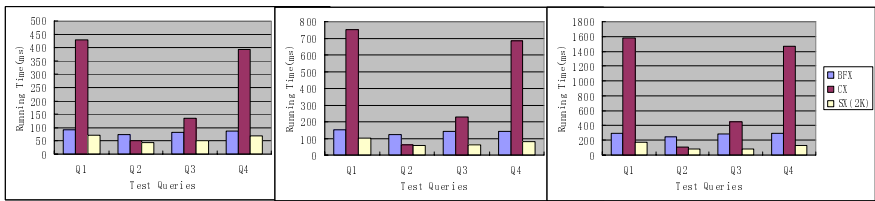


Fig. 8. Elements scanned

5.2 Number of Elements Scanned

Figure 8 shows the total number of elements scanned for XR-Tree and our technology. In this experiment, we only demonstrate the experimental results for XMark 1.0 due to the space limitation of this paper.

It can be seen from figure 8 that our filter strategy leads to the least number of elements compared to the XR-Tree index. The benefit gets more obvious for Q1 and Q4. The simple reason is that the pairs of 1:1 and 1:0 relations are involved in Stack-Join method. As a result, many ancestor nodes are scanned to verify the relation between two node sets.



(a) XMark 1.0

(b) XMark 2.0

(c) XMark 4.0

Fig. 9. Difference of running time

5.3 Running Time

To study the running time, we conduct two sets of experiments. In the first set of experiments, we assume that the block size of SX-Filter is 64K bits, and evaluate the running time for these methods. In the latter set of experiments, we investigate the

running time of SX-Filter approach on varying the block size. The running time is the average of 10 times run.

Figure 9 shows the difference of running time. It can be observed that the SX-Filter performs better than other two methods. The reason is that the bit vectors are sparse. As a result, more blocks are skipped, in which there is no need to verify the exact positions for elements. It is also observed that CX-Filter performs worse for most queries. This is because many comparison operations are applied to verify the exact positions to be evaluated.

From figure 10(a), we notice that the increase of block size leads to the decrease of the running time when block size is lower than 64K bits, but leads to the increase along with the crescent of block size.

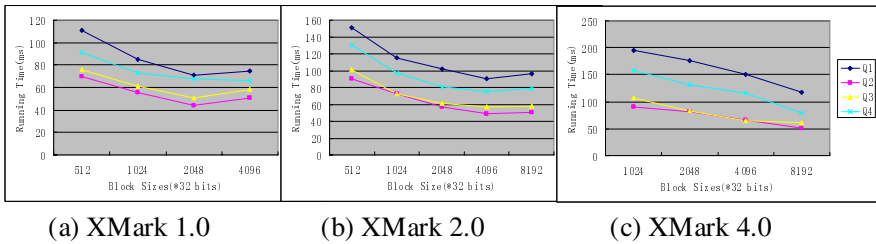


Fig. 10. Time cost of SX-Filter on varying block sizes

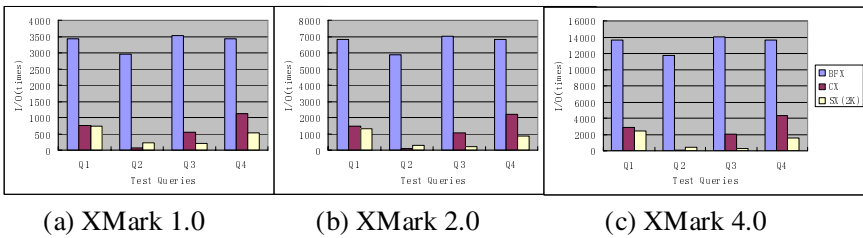


Fig. 11. Difference of I/O cost

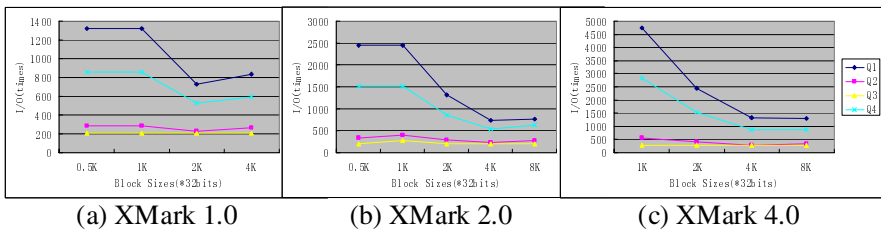


Fig. 12. I/O cost of SX-Filter on varying block sizes

5.4 Numbers of Page Accesses

This metrics measures the performance of algorithms in terms of I/O cost. Figure 11 shows the difference of I/O cost. It can be observed that the SX-Filter and CX-Filter are all significant lower than BFX-Filter. This is because the BFX-Filter reads all the bit vectors, and I/O cost is wasted. Note that SX-Filter overperforms CX-Filter. This is owing to the fact that the exact position can not be verified in header files, and only brute force scan can be applied to these files.

The I/O cost of SX-Filter algorithm on varying the block sizes is summarized in figure 12. From figure 12 we can see that the increase of block size leads to the decrease of the I/O cost when the block size is lower than some critical value, but leads to the increase along with the crescent of block size. It is similar to the running cost on varying the block sizes. To sum up, the running cost of SX-Filter is mainly contributed by I/O times.

6 Conclusions

In this paper, we propose a novel filter strategy for reducing the number of candidate nodes participated in XML query processing. Based on schema information, some definitely useless nodes are skipped. The experimental results show that our method is effective, scalable and efficient. To the best of our knowledge, no previous work has addressed to reduce the number of candidate nodes based on XML schema information.

References

1. H.Jiang, H.Lu, W.Wang, and J.X.Yu. Holistic Twig Joins on Indexed XML Documents. In VLDB, pages 273-284, 2003.
2. S.-Y.Chien, Z.Vagena, D.Zhang, V.Tsotras, and C.Zaniolo. Efficient Structural Joins on Indexed XML Documents. In VLDB 2002, pp: 263-274.
3. H.Jiang, H.Lu, W.Wang, and B.C.Ooi. XR-Tree: Indexing XML Data for Efficient Structural Joins. In ICDE 2003, pp: 253-264.
4. Q.Li, B.Moon. Indexing and Querying XML Data for Regular Path Expressions. In VLDB 2001, pp: 361-370.
5. D.Srivastava, S.Al-Khalifa, H.V.Jagadish, N.Koudas, J.M.Patel, and Y.Wu. Structural Joins: A Primitive for Efficient XML Query Pattern Matching. In ICDE 2002, pp:141-152.
6. Y.Wu, J.M.Patel, and H.V.Jagadish. Structural Join Order Selection for XML Query Optimization. In ICDE 2003, pp: 443-454.
7. C.Zhang, J.F.Naughton, D.J.Dewitt, Q.Luo, and G.M.Lohman. On Supporting Containment Queries in Relational Database Management Systems. In SIGMOD 2001, pp: 425-436.
8. N.Bruno, D.Srivastava, and N.Koudas. Holistic Twig Joins: Optimal XML Pattern Matching. In SIGMOD, pages 310-321, 2002.
9. H.Jiang, H.Lu, W.Wang. Efficient Processing of XML Twig Queries with OR-Predicates. In SIGMOD, pages 59-70, 2004.
10. J.Lu, T.Chen, T.W.Ling. Efficient Processing of XML Twig Patterns with Parent Child Edges: A Look-ahead Approach. In CIKM, 533-542, 2004.
11. H.K.T.Wong, J.Z.Li, F.Olken, D.Rotem and L.Wong. Bit Transposition for Very Large Scientific and Statistical Databases. In VLDB, pages 304-311, 1986.

An Effective and Efficient Approach for Keyword-Based XML Retrieval*

Xiaoguang Li, Jian Gong, Daling Wang, and Ge Yu

School of Information Science and Engineering, Northeastern University
Shenyang 110004, P.R.China
xiaouangli@163.com, {jiangong, dlwang, yuge}@mail.neu.edu.cn

Abstract. IR-style keyword-based search on XML document has become the most common tool for XML query, as users need not to know the structural information of the target XML document before constructing a query. For a keyword-based search engine for XML document, the key issue is how to return some sets of meaningfully related nodes to user's query efficiently. An ordinary solution of current approaches is to store the relationship of each pair of nodes in an XML document to an index. Obviously, this will lead to serious storage overhead. In this paper, we propose an enhanced inverted index structure (PN-Inverted Index) that stores path information in addition to node ID, and import and extend the concept of LCA to PLCA. Efficient algorithms with these concepts are designed to check the relationship of arbitrary number of nodes. Compared with existing approaches, our approach need not create additional relationship index but just utilize the existing inverted index that is much common for IR-style keyword search engine. Experimental results show that with the promise of returning meaningful answers, our search engine offers great performance benefits. Although the size of the inverted index is increased, the total size of indices of search engine is smaller than the existing approaches.

1 Introduction

As the evolution of the Internet, XML has become the standard of data publishing and exchanging. Now, more and more people often need to get information from XML documents. How to retrieve information efficiently and effectively became a hot-point in this research area.

Traditionally, the research works of XML document retrieval can be divided into two taxonomies: structural query and keywords-based search. XPath [9] and XQuery [10] are the generally accepted standards of the former. For structural query, user usually needs to know structural information of the target XML document before constructing a query. This kind of query can reveal what he queries about indeed. However, most of XML documents in the real world are lack of DTD or XSD. Even though such structural information exists, the structure maybe varied with XML

* Supported by the National Natural Science Foundation of China(60173051), and the Teaching and the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institution of the Ministry of Education of China.

document of the same content. In such a case user need write the query expression for every document. In contrast, a query of keywords-based search is easy to be constructed even by naïve users, as it does not need any structural information of the target document. Keywords-based search for XML is quite different from that for HTML or text files in that the search engine does not return a whole document, but return meaningful fragments consisted of some related nodes in the target XML document.

The basic issue to keyword-based search is how to determine which sets of nodes that satisfy the query are meaningful. Now, most of approaches to this issue are based on the concept of Lowest Common Ancessor (LCA), and almost every IR-style keyword search engine needs an inverted index to store information corresponding to every keyword in the target documents. For each keyword, traditional inverted index stores the node identifier, together with the document identifier if there is more than one file in the repository. This kind of inverted index has been proved to be much efficient for traditional keyword searching and ranking without checking the meaningfulness of the returned nodes. However, returning the LCA of nodes directly to user without meaningfulness checking, as Meet [11] does, will clearly lead to poor retrieval precision. On the other hand, if we add meaningfulness checking step after getting the LCA, the traditional inverted index is not suitable for efficient checking as there is no structural information of the node stored in the index. In this case, the search engine has to create additional index to store the relationship for each pair of nodes, or provide an online algorithm, as XSearch [1] does. Inevitably, the former will increase space complexity, and the latter will be time-consuming.

In this paper, we propose an enhanced inverted index structure, PN-Inverted Index, which stores path information of each keyword in addition to node id. The concept of LCA is imported and extended to be PLCA (Lowest Common Ancessor of Label Path), and efficient algorithms with these concepts are designed to check if arbitrary numbers of nodes in an XML document are meaningfully related. Compared with existing approaches, our approach need not create additional relationship index but just utilize the existing inverted index that is much common for IR-style keyword search engine, and all indices in this paper can be efficiently created after parsing the XML document only once. Experimental results show that with the promise of returning meaningful answers, our approach offers great performance benefits. Although the size of inverted index is increased somewhat, the total size of indices is much smaller than the existing approaches.

The rest of the paper is organized as follows. Section 2 introduces the background knowledge of this paper, discusses major related works and our motivation. Section 3 introduces some definitions and concepts we used, and proposes the concept of PLCA and PLCA rule to check the relationship of nodes. Section 4 describes the structure of PN-Inverted Index, and gives the query evaluation algorithm. The experimental results and analysis are presented in Section 5. Section 6 concludes the whole paper.

2 Preliminaries and Related Work

Let $Q = \{q_1, q_2, \dots, q_k\}$ be query, where q_i is a keyword. Let N_Q be the set of nodes that satisfying Q , i.e. $N = N_Q^1 \times N_Q^2 \times \dots \times N_Q^k$, where N_Q^k is the set of node that satisfying

keyword q_k . The basic issue is how to determine which elements of N_Q are meaningful.

For example, the following XML document fragment is an excerpt from XMark [16]. Suppose a query $Q = \{computer, white\}$ on the XML document in Fig.1. As one of answers, the first `<name>` node and second `<color>` node will be returned. However, these two nodes are not meaningfully related to each other as they are describing the different items and should not be returned.

Towards this issue, many concepts and approaches are proposed, such as Meet [11], XSearch [1] and Schema-Free XQuery [2], but most of approaches are based on the concept of Lowest Common Ancestor. In this section, firstly we will introduce this concept and describe a widely accepted rule of checking the relationship of nodes. Secondly, some related works are discussed.

```

<Asia>
  <item>
    <name> computer </name>
    <color> black </color>
  </item>
  <item>
    <name> car </name>
    <color> white </color>
  </item>
</Asia>

```

Fig. 1. An XML document

2.1 Lowest Common Ancestor

Definition 1. An XML document is a rooted, ordered tree $T = (N, E, r)$ with nodes N and edges $E \subseteq N \times N$ and a distinguished node $r \in N$, the root node. The set of nodes $N = NE \cup NV$, where NE is the set of elements, also called the set of *inner nodes*, and NV is the set of values, also called the set of *leaf nodes*.

For a tree T , let $Label: NE \rightarrow String$ be the function that assigns the label to the node of NE , and let $Value: NV \rightarrow anyType$ [17] be the function that assigns the value to the node of NV . A *path* p in a tree T is a sequence of nodes u_1, u_2, \dots, u_m such that for every pair u_i, u_{i+1} of consecutive nodes there is an edge $(u_i, u_{i+1}) \in E$. A node u is called an ancestor of an node v iff there is a path $u = u_1, u_2, \dots, v = u_m$, where $m > 1$ that lead from u to v . If $m = 1$, u is called the parent of v (and v is the child of u).

Definition 2. (Ancestor-or-self) A node N_a is said to have a *Ancestor-or-self* relationship with N_d if it is an ancestor of N_d or is equal to N_d , denoted as *Ancestor-or-self* $(N_a, N_d) = true$.

Definition 3. (LCA of two nodes) For nodes $u_1, u_2 \in N$, a node $u \in N$ is the LCA of u_1 and u_2 if and only if:

- *Ancestor-or-self* $(u, u_1) = true$, *Ancestor-or-self* $(u, u_2) = true$, and
- $\forall u' \in N$, if *Ancestor-or-self* $(u', u_1) = true$ and *Ancestor-or-self* $(u', u_2) = true$, then *Ancestor-or-self* $(u, u') = true$

u is denoted as $LCA(u_1, u_2)$.

Based on the concept of LCA, a widely accepted rule of checking the relationship of nodes is to utilize the relational subtree as the context and lookup whether or not a pair of nodes shares the same label. This rule is proposed firstly in XSearch and has

been proved to be suitable for most situations. Here, this rule is called LCA rule, and introduced in brief.

LCA Rule: given nodes u_1 and u_2 in the XML document, the relational subtree of u_1 and u_2 (denoted as T_{u_1, u_2}) is a subtree with root $LCA(u_1, u_2)$ and all the nodes on the path from $LCA(u_1, u_2)$ to u_1 and u_2 , respectively. Then u_1 and u_2 are related meaningfully if the following conditions hold:

1. T_{u_1, u_2} does not contain two distinct nodes u and v , where $Label(u)=Label(v)$.
2. If there are two distinct nodes u and v in T_{u_1, u_2} , where $Label(u)=Label(v)$, then $u=u_1$ and $v=u_2$, or $v=u_1$ and $u=u_2$.

2.2 Related Work

Numerous works have been done about query on XML document. Traditionally, research work in this area has been following one of the two paths: the structured query approach and the keyword-based approach. XPath[9] and XQuery [10] are the generally acknowledged standard of the former, while the latter class has several recent suggestions, including Meet [11], XRANK [8], Schema-Free XQuery [2] and XSearch [1].

Meet operator returns the LCA as query answer. However, it does not consider the meaningfulness of the LCA, which will lead to poor precision especially while querying heterogeneous XML documents repository. XRANK has a ranking mechanism and it returns document fragments as answers. However, XRANK just return the most special fragment as answers, of which parts maybe semantically unrelated. Schema-Free XQuery put forwards a concept of meaningful lowest common ancestor based on the concept of LCA. It shows a promising precision of query that outperforms other approaches. But Schema-Free XQuery uses and extends XQuery as its query language, which is not as easy to write a query as IR-style keyword search for naïve users. Furthermore it requires accessing all the nodes with the same entity type, even a query only contains keyword, not any element. This feature increases time-complexity in a great degree.

XSearch is a keyword-based semantic search engine, where the interconnection relationship is introduced to determine if two nodes in an XML document are meaningfully related to each other through the rule mentioned in Section 2.1. XSearch developed a syntax for search queries that is suitable for a naïve user and facilitates a fine-granularity search. It provide an offline index to record whether each pair of nodes in a document is meaningful related, but this offline index leads to serious storage overhead as the index size is much larger than the original file. XSearch also notice this point and proposes an online indexing method to make the index size smaller, while increases greatly the cost of evaluating query, especially, when deal with query on XML documents with deep hierarchical structure.

Some query languages and search engines are proposed to support keyword searching and result ranking on XML document, such as XIRQL [12], XXL [13]. EquiX [14] proposed an advanced ranking method on returned answer set which is similar to page-rank. We did not add any ranking mechanism to our search engine. However, all these ranking methods can be easily integrated to our search engine.

After all, all the previous work could not efficiently perform IR-style keyword search on XML document. They either implement with high time and space complexity, or not consider meaningfulness of the query answer at all. In this paper, we achieve this goal by means of a novel concept of lowest common ancestor of label path (PLCA).

3 LCA of Label Path

We found the rule mentioned in Section 2.1 can give appropriate result for most documents and adopt it as the meaningfulness checking criterion in our approach. Our approach can efficiently determine whether or not two nodes are meaningfully related without storing any relationship of pair of nodes in advanced. This section discusses node encoding method which is absolutely necessary to our approach, and then gives a theorem of relationship determination, which is basement of the following algorithm of relationship determination.

3.1 XML Encoding

There are many encoding methods for XML document, such as absolute region code [3], relative region coordinate [4], region-based code [5], PBiTree coding [6] and XR-tree [7].

In this paper, we use Dewey encoding of node id. The main reason is that Dewey encoding captures the relationship of ancestor and descendant information. As discussed in the following, this feature is very helpful to determine the relationship of nodes. Due to the limitation of space, the detail of Dewey id is referred to [8], here we just give an example as illustrated in Fig. 2. Let $|●|$ be the length of id or path, for example $|0.0.1|=3$, $|regions.Asial|=2$.

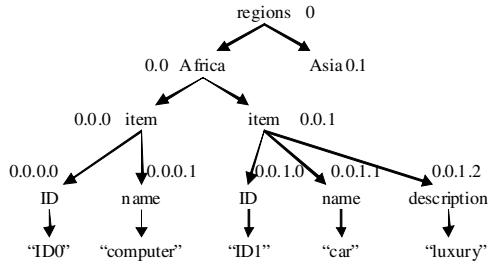


Fig. 2. An illustration of Dewey id

Property 1. For nodes $u_1, u_2 \in N$, if $Ancestor(u_1, u_2) = true$, then the Dewey id of u_1 is a prefix of id of u_2 .

Certainly, other encoding also can be applied into our approach if they can get the identifier of LCA and the depth from root to LCA. Here Dewey id is used just for its simplicity of getting the identifier of LCA.

3.2 Relationship Determination with LCA of Label Path

Definition 4. (Label path) A label path $lp = l_1.l_2...l_n$ is a sequence of label names of a path $u_1.u_2,...,u_n$. A prefix path lp' of lp is a sub-sequence of lp , where lp' has the same beginning with lp and $|lp'| \leq |lp|$.

Definition 5. (LCA of two label paths) For label path lp_1 and lp_2 , a label path $lp = l_1l_2\dots l_m$ is said to be the *Common Path* of lp_1 and lp_2 if and only if:

- lp is the prefix path of both lp_1 and lp_2 , and
- $\forall lp', lp'$ is a prefix path of both lp_1 and lp_2 , then lp' is the prefix path of lp and then l_m is the LCA of lp_1 and lp_2 , denoted as $PLCA(lp_1, lp_2)$.

Property 2. For nodes $u \in N$, let id and lp be the Dewey id of u and label path from the root to u , respectively, then $lid = |lp|$.

Lemma 1. For a tree T , given nodes u_1 and u_2 , let lp_1 and lp_2 be the label path from the root of T to u_1 and u_2 respectively, suppose lp is the common path of lp_1 and lp_2 , and $u = LCA(u_1, u_2)$, then $|lp| \geq lid$ of u .

Proof. Suppose lid of u is k and id of u is i_1, i_2, \dots, i_k , then according to property 1, the id of u_1 and u_2 is $i_1, i_2, \dots, i_k, i_{k+1}, \dots, i_m$ and $i_1, i_2, \dots, i_k, i'_{k+1}, \dots, i'_n$, respectively. According to property 2, the corresponding label paths from u_1 and u_2 are supposed as $r.l_2.l_3\dots l_k.l_{k+1}\dots l_m$ and $r.l'_2.l'_3\dots l'_k.l'_{k+1}\dots l'_n$, respectively. Obviously, for any l_i and l'_i , $i=2\sim k$, $l_i=l'_i$. If $l_{k+1}\dots l_m$ and $l'_{k+1}\dots l'_n$ do not share any common prefix, then lp is $root.l_2.l_3\dots l_k$, and then $|lp|=lid$ of u . Else let the length of common prefix is k' ($k' > 0$), then $|lp|=k+k'$, and then $|lp| > lid$ of u .

Now, based on the concept of PLCA, we give a rule of checking the meaningful relationship of nodes, called PLCA rule.

PLCA rule: for a tree T , given nodes u_1 and u_2 , lp_1 and lp_2 is the label path from the root of T to u_1 and u_2 , respectively, suppose lp is the common path of lp_1 and lp_2 , and $u = LCA(u_1, u_2)$. u_1 and u_2 are meaningfully related if the following conditions holds:

- $|lp|=lid$ of u
- There are not two distinct label with the same name in the set of $(lp_1-lp) \cup (lp_2-lp)$
- The only two distinct label with the same name in the set of $(lp_1-lp) \cup (lp_2-lp)$ are the ends of lp_1 and lp_2

Theorem 1. PLCA rule is equivalent to LCA rule.

Proof. According to Lemma 1, the label of u must be contained in lp . If $|lp| \neq lid$ of u , then there are at least two nodes in $T|_{u_1, u_2}$ with the same label of $PLCA(lp_1, lp_2)$. Obviously the condition 2 and 3 consist with the condition 1 and 2 of LCA rule. So if nodes u_1 and u_2 satisfy PLCA rule then also satisfy LCA rule, vice versa.

Theorem 2. The time complexity of implementing PLCA rule is $O(c+N_{plca}!)$, where c is a constant time of verifying condition 1 and 3, $N_{plca} = |(lp_1-lp) \cup (lp_2-lp)|$.

Theorem 2 is intuitive and its proof is omitted due to the limitation of space.

4 PN-Inverted Index and Query Algorithm

Based on the concept of PLCA and Theorem 2, we design a proper index structure, PN-Inverted Index, and develop an online algorithm to make the query progress effective and efficient.

4.1 Index Structure

In our search engine, we design an enhanced inverted index, PN-Inverted Index. Here, The PN-Inverted Index created only for a single document is discussed, and then the document id is omitted. The support to a document repository is straight forward. The basic idea of PN-Inverted Index is to store the label path for each node from root to this node in addition to its id. As shown in next section, this feature turns online checking into reality. PN-Inverted Index consists of two parts as shown in Fig. 3.

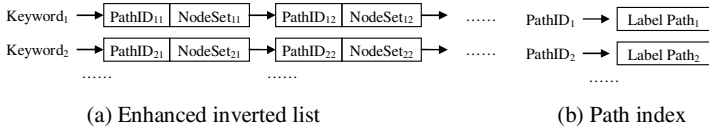


Fig. 3. PN-Inverted Index

The first part is an enhanced inverted index as shown in Fig. 3 (a). In each entry of the inverted index, for a keyword, ids of all the nodes that containing this keyword, and for such each node, the label path from root to this node are stored. Here two optimizations are made to reduce the size of index. Firstly, although the original XML document may be of a large size and lots of nodes, the number of label paths is usually small, as this number is determined by the schema of the document but have nothing with the document size [15]. So in the our inverted index, for each entry, those nodes with the same label path are grouped into together, called NodeSet, and only one label path is stored for all these nodes. Clearly this would make the size of inverted index smaller, especially when search engine support query on the label name of nodes. Secondly, to further compress the inverted index, an id of label path, called PathID, is created and stored instead of the label path itself, and then Path Index is built to store the mapping of PathID and label path, as shown in Fig.3 (b). Note that label paths for those nodes which have at least one text node as their child are created, as label paths of other nodes would never appear in this inverted index.

Theorem 3. The space complexity of PN-Inverted Index is $O(n_w n_{lp} + n_w n_{lp} \bar{n}_u + n_{lp})$, where n_w is the number of keywords, n_{lp} is the number of label paths, \bar{n}_u is the average number of nodes for each label path.

Proof. For each entry of inverted index, its space complexity is $O(n_{lp} + n_{lp} \bar{n}_u)$, and for path index, its space complexity is $O(n_{lp})$, so the total space complexity is $O(n_w n_{lp} + n_w n_{lp} \bar{n}_u + n_{lp})$.

However, all the relationships of nodes should be recorded in the traditional inverted index, such as XSearch, the space complexity in this case is $O(n_w n_{lp} \bar{n}_u + |N|^2)$, where N is the set of all nodes in an XML document. Generally speaking, $|N|$ is usually very large, in such case the space complexity of $O(|N|^2)$ is much greater than that of $O(n_w n_{lp} + n_{lp})$. Algorithm 1 is the algorithm of PN-Inverted

Index creating. Note that this algorithm create inverted index only for the keywords in each value node, not for the labels of nodes, but it is easy to be extended to support it.

Algorithm 1: CreateIndex

Input: an XML document

Output: PN-Inverted Index

1. while (parser.readNode(*node*) != NULL)
2. if (*node* is the start of an element and $node \in NE$)
3. if (!*stack*.isEmpty) *stack*[*top*].*childCount* ++;
4. *childCount* = 0; *label* \leftarrow getCurrentNodeLabel();
5. *stack*.push(*label*, *childCount*);
6. if ($node \in NV$)
7. *curID* \leftarrow getIDfromStack();
8. *curLabelPath* \leftarrow getLabelPathfromStack();
9. *curPathID* \leftarrow pathIndex.getPathID(*curLabelPath*);
10. for each word in the text
11. *invertedIndex*.addEntry(*word*, *curID*, *curPathID*)
12. if (*node* is the end of an element and $node \in NE$) *stack*.pop();
13. return;

Algorithm 1 creates the index meanwhile computing the Dewey id for current node. The parser reads in every node in the XML document in pre-order (line 1) and access the document only once. When the access node is the start of an element belonging to the set of NE , such as “<regions>”, the algorithm gets the label of node and push it into stack (line 2~5). If the access node is the end of an element, then pop the top of stack (line 12). So the sequence of stack from the bottom to top is proved to be current label path with these processing. If the node is a value node, the algorithm traverses the stack from bottom to top to compute id and label path of current nodes, and then add an entry for each word (line 6~11). The addEntry (line 11) function adds a given word to the corresponding entry and cluster according to given node id and path id.

Theorem 4. The time complexity of algorithm 1 is $O(N_v + \log_2^{n_w} + N_v \log_2^{n_p})$.

Proof. Each input node of value is pushed into the stack or popped from the stack at most once. Since the stack operation falls into the one of these constant time operations, the time complexity is $O(N_v)$, where N_v is the number of value nodes. Suppose the total number words in the document is n_w , the total number used to add keywords to the entry of inverted index is $\log_2^{n_w}$. Then suppose each entry contain at most n_p different label paths, the number used to add node id to the corresponding label path is $N_v \log_2^{n_p}$. So, the total time complexity is $O(N_v + \log_2^{n_w} + N_v \log_2^{n_p})$.

4.2 Query Evaluation

While checking meaningful relationship of arbitrary number of nodes, *all-pairs* relationship [1] is adopted, which means every pairs of nodes should be related

meaningfully, and leads to a higher precision than *star-related* relationship, which means that all nodes are related with a *star center* node.

4.2.1 Naïve Implementation

For a keyword-based query $Q=\{q_1, q_2, \dots, q_k\}$, the straight forward implementation is that firstly, the search engine lookups each of the k keywords in the inverted index and returns k set of nodes, each corresponding to one keyword, then computes the Cartesian product of these k sets, of which each element is a potential answer, finally executes the meaningfulness checking based on *PLCA* rule on each potential answer and adds those meaningful ones into final answer set. The problem with this implementation is that the intermediate result set may grow to be of very large size. This kind of implement suffers from high complexity of both space and time, even though there may be just a piece of potential answers are meaningful.

Theorem 5. The time complexity of naïve implementation is $O(c_{\max} k(k-1) \prod_{i=1}^k n_i)$, where n_i is the number of returned nodes corresponding to each keyword, c_{\max} is the maximum time of verifying *PLCA* rule.

Proof. There are $\prod_{i=1}^k n_i$ potential results after the Cartesian product calculated. The number of verifying *PLCA* rule for each result is $k(k-1)/2$. So the total time complexity is $O(c_{\max} k(k-1) \prod_{i=1}^k n_i)$.

4.2.2 Efficiently Query Evaluation

Obviously naïve algorithm is simple, but expensive. To improve the efficiency, we develop another algorithm. Since *all-pairs* relationship requires every pairs of nodes should be related meaningfully, the basic idea of this algorithm is that if two nodes are meaningless, then any answers containing these nodes inevitably lead to a meaningless result at last, and then prune them as early as possible. The algorithm is as follow.

Algorithm 2: EvaluateQuery

Input: a query $Q=\{q_1, q_2, \dots, q_k\}$

Output: meaningful sets of nodes

1. for each $q_i \in Q$ in query
2. get *PNPairSet* that is a set of node id and path id from inverted index with q_i ;
3. add *PNPairSet* into *PNPairSetN*; // *PNPairSetN* is an array of *PNPairSet*
4. let the intermediate Cartesian product be I ;
5. for each *PNPairSet* in *PNPairSetN*
6. $I \leftarrow I \otimes \text{PNPairSet}$;
7. prune meaningless results in I according to *PLCA* rule;
8. return I ;

Note that the \otimes operator (line 6) computes Cartesian product of two set. Line 7 prunes meaningless results immediately after getting the intermediate results. While checking meaningfulness between I and new set of nodes, all nodes in I have been meaningfully related to each other as we have checked them in the previous step. If the size of query result is small, then a smaller size of intermediate result in this

algorithm than in algorithm 1 maybe obtained early, and this feature will absolutely decrease the time complexity.

Theorem 6. The time complexity of algorithm 2 is $O(c_{\max} n_{\max} \sum_{i=1}^k n_i (i-1))$, where n_{\max} is the maximum number of intermediate result.

Proof. The maximum number of intermediate result at the i^{st} is $n_{\max} n_i$, and at this time the number of verifying *PLCA* rule is $i-1$, so the total time complexity is $O(c_{\max} n_{\max} \sum_{i=1}^k n_i (i-1))$.

5 Performance Experiments

5.1 Experimental Environment

We performed extensive experiments with our search engine, which was implemented in C++. The experiments were carried out on a Pentium 4, with a CPU of 2.8G and 512M RAM, running Windows XP operation system. When parsing an XML document, MSXML3.0 and SAX model are

adopted. XMark [16] is used as the data set with factor varies from 0.001 to 0.1, as a result the size of XML document varies from 116K to 11.5M and the detail of these dataset is shown in Table 1. We constructed 300 queries with the keywords random selected from all the words appeared in the document, and the number of keywords of query varies from 2 to 4.

The experimental results are compared with XSEach [1] as it is a famous keyword search engine on XML documents. We compare the size of index, the indexing time and query responding time with it. Note that as proofed before, the meaningfulness judge criteria of ours is equal to those used in XSearch, so the precision and recall is also equal to theirs and we do not give the result here.

5.2 Result Analysis

In Fig. 4, the comparison of the size of index under different approaches with different size of XML documents is presented. Observe that our total index size is just a few bigger than traditional inverted index. The total size of XSearch is small when online indexing strategy is used, while it becomes very large when offline indexing is created, which can be 100 times of the original document.

In Fig.5, we present the result of indexing time. The indexing time of traditional inverted index and XSearch without interconnection index is the faster than ours, but our indexing time is faster than XSearch with interconnection index. We note that the indexing time of interconnection in XSearch is short when the structure of XML document is flat. But when the depth grows, the indexing process becomes very time consuming as it need to recursive compute many times. However, the indexing time

Table 1. Summary of dataset

Dataset id	Size(kb)	Number of node
1	116	1729
2	1155	17132
3	11597	167865

of our index has nothing with the structure of the document, but merely determined by the original document size.

For each approach, we run 300 queries with the number of keyword varying from 2 to 4. All the keywords in the queries are drawn randomly from the set of keywords in the document. From Fig.6, XSearch with interconnection index is faster than our approach, while much slower without interconnection index. In addition, as our approach prunes meaningless answers as early as possible, which makes the intermediate not grow too large, and the query time not increases exponentially with the document size.

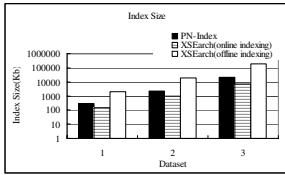


Fig. 4. index size

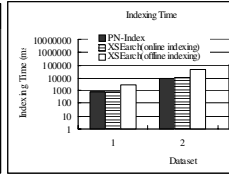


Fig. 5. indexing time

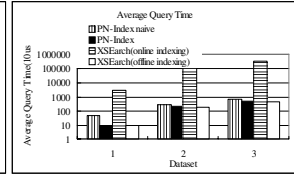


Fig. 6. average query time

6 Conclusion

In this paper, the concept of LCA, a widely accepted concept, is introduced and extended to the concept of PLCA, and an enhanced index and algorithm of query evaluation supporting this concept are developed to efficiently execute keyword-based query with promise of query precision, to efficiently and effectively retrieve meaningful results for keyword-based XML query. Main contributions in this paper are:

- A new concept of PLCA and a PLCA rule of checking the meaningful relationship of nodes, where PLCA rule is equivalent to LCA rule. These innovations turn on-line query evaluation into practice.
- An enhanced index, PN-Inverted Index, whose size is compressed much more than the traditional approaches, and the creating time is linear with the size of document.
- An efficient algorithm of query evaluation, which can prune meaningless results early, and then decreases the time complexity of evaluation greatly.

References

1. S. Cohen, J. Mamou, Y. Kanza, Y. Sagiv. XSearch: a semantic search engine for xml. Proc. of VLDB, 2003
2. Y. Li, C. Yu, H. V. Jagadish. Schema-free XQuery. Proc. of VLDB, 2004
3. C. Zhang, J. Naughton, D. DeWitt, Q. Luo, G. Lohman. On supporting containment queries in relational database management systems. Proc. of SIGMOD, 2001
4. D. D. Kha, M. Yoshikawa, S. Uemura. An XML indexing structure with relative region coordinates. Proc. of ICDE 2001.

5. Q. Li, B. Moon. Indexing and querying XML data for regular path expressions. Proc. of VLDB 2001.
6. W. Wang, H. Jiang, H. Lu, J. X. Yu. PBiTree coding and efficient processing of containment joins. Proc. of ICDE, 2003.
7. H. Jiang, H. Lu, W. Wang, B. C. Ooi. XR-Tree: indexing xml data for efficient structural joins. Proc. of ICDE, 2003.
8. L. Guo, F. Shao, C. Botev, J. Shanmugasundaram. XRank: ranked keyword search over xml documents. Proc. of SIGMOD, 2003.
9. A. Berglund, S. Boag, D. Chamberlin, M. F. Fernandez, M. Kay, J. Robie, J. Simeon. XML path language (XPath) 2.0. W3C working draft. Available from <http://www.w3.org/TR/xpath20/>, 2002
10. S.Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, J. Simeon. XQuery 1.0: an xml query language. W3C working draft. <http://www.w3.org/TR/xquery/>, 2003
11. A. Schmidt, M. Kersten, M. Windhouwer. Querying xml document made easy: nearest concept queries. Proc. of ICDE, 2001
12. N. Fuhr and K. Grobjoam. XIRQL: a query language for information retrieval in XML document. Proc. of SIGIR, 2001.
13. A. Theobald and G. Weikum. The index-based XXL search engine for querying XML data with relevance ranking. Proc. of EDBT, 2002.
14. S. Cohen, Y. Kanza, Y. Kogan, W. Nutt. Y. Sagiv and A. Serebrenik. EquiX: a search and query language for XML. Proc. of JASIST, 2002.
15. B. Choi. What are real dtds like? Proc. of the Fifth International Workshop on Web and Database (WebDB), 2002.
16. XMark. <http://monetdb.cwi.nl/xml/index.html>, 2003.
17. W3C. XML schema, <http://www.w3.org>, 2003.

Subgraph Join: Efficient Processing Subgraph Queries on Graph-Structured XML Document*

Hongzhi Wang^{1,2}, Wei Wang¹, Xuemin Lin¹, and Jianzhong Li²

¹ University of New South Wales, Australia
wangzh@hit.edu.cn, {weiw, lxue}@cse.unsw.edu.au

² Harbin Institute of Technology, Harbin, China
lijz@mail.banner.com.cn

Abstract. The information in many applications can be naturally represented as graph-structured XML document. Structural query on graph structured XML document matches the subgraph of graph structured XML document on some given schema. The query processing of graph-structured XML document brings new challenges.

In this paper, for the processing of subgraph query, we design a subgraph join algorithm based on reachability coding. Using efficient data structure, subgraph join algorithm can process subgraph query with various structures efficiently.

1 Introduction

XML has become the *de facto* standard for information representation and exchange over the Internet. XML data has hierarchy nesting structure. XML data is often modeled as a tree. However, XML data may also have IDREFs that add additional relationship to XML data. With such property, XML data also can be represented in graph structure. In many applications, data can be modeled as a graph more naturally than a tree.

Of course, graph structured XML document can be represented in tree structure by duplicate the element with more than one incoming paths. But it will result in redundancy.

Query languages are proposed for XML data. XQuery [4] and XPath [6] are query language standards for XML data. Structure query on graph structured XML data has more power. Further than branching query on tree structured XML data, structure query on graph-structured XML data can request subgraphs matching the general graph modeled schema described query.

Query processing on graph structured XML data brings new challenges:

- More complex query can be defined on graph-structured XML data. The query can be also graph-structured to retrieve a subgraph of an XML document. The schema of the subgraph can be various, possibly including nodes

* This work was partially supported by UNSW FRG Grant (PS06863), UNSW Goldstar Grant (PS07248) and the Defence Pre- Research Project of the Tenth Five-Year-Plan of China no.41315.2.3.

with multiple parents or circle. Existing method cannot process such query efficiently.

- One way to processing structural query on XML data is to encode the nodes of graph with some labelling scheme. With the code, the structure relationship such as parent-child or ancestor-descendant can be judgment quickly. In query processing on tree structured XML, it is a well-studied problem. But all existing labeling scheme of XML representations and query processing methods are based on tree model. They can not be applied on graph-structured XML data directly.
- Another kind query processing methods for XML is to use structural index such as 1-index[15], F&B index[13] to accelerate the query processing. But the structural index of graph structured XML document has many nodes. It is not practical to use structural index directly to process query on graph structured XML. For example, the number of nodes in F&B index of tree structured 100M XMark document has 436602 nodes while the number of nodes in F&B index of graph structured 100M XMark document has 1.29M nodes [13].

Using label to represent the relationship between nodes is a practical method to process query on graph-structured XML data. With well-designed labeling, the structural relationship between two nodes can be determined efficiently without accessing any other node. In this paper we use an extension of the code in [16] as reachability code.

To process the complex queries with a graph schema on graph-structured, we design a novel subgraph join algorithm based on the reachability code. In order to support the overlapping of intervals in the coding, we design a data structure *interval stack*. Subgraph join algorithm uses a chain of linked interval stacks to compactly represent partial results. Subgraph join algorithm can be used to process subgraph query with both adjacent and reachability relationship.

The contributions of this paper can be summarized as follows:

- We use duplication to make the coding possible to be storage in relation or apply sorted based join algorithms on.
- We present efficient graph structural join algorithms and efficient data structure, interval stack, to support join.
- We present subgraph query, a novel kind of structure query using general graph as matching schema. To process subgraph query, we design a novel subgraph join algorithm. It processes subgraph query efficiently.

The rest of the paper is organized as follows: Section 2 introduces some background knowledge. Data preprocessing and subgraph join algorithm are presented in Section 3. We present our experimental results and analysis in section 4. Related work is described in Section 5. We conclude the paper in Section 6

2 Preliminaries

In this section, we briefly introduce Graph-structural XML model and some terms used in this paper.

2.1 Data Model

XML data is often modeled as a labelled tree: elements and attributes are mapped into nodes of graph; directed nesting relationships are mapped into edges in the tree. A feature of XML is that from two elements in XML document, there may be a IDREF representing reference relationships [23]. With this feature, XML data can be modeled as a labelled digraph: elements and attributes are mapped into nodes of graph; directed nesting and reference relationships are mapped into edges in the graph. An XML fragment is shown in Fig 1(b). It can be modeled as the graph shown in Fig 1(b). It is noted the graph in Fig 1(b) is not a DAG.

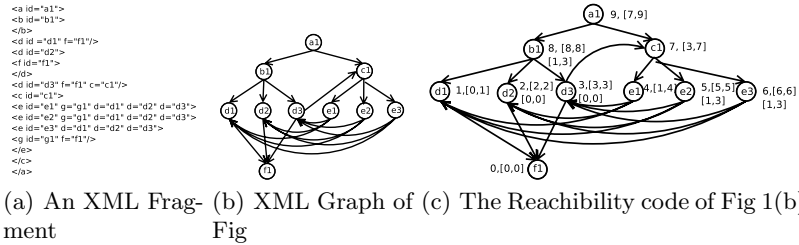


Fig. 1. An Example of Graph-structured XML

In a graph, a node without incoming edge is called *source*. A node without outgoing edge is called *sink*.

2.2 Subgraph Query

In graph-structured XML, the parent-child and ancestor-descendant relationship should be extended. In [13], the idref edges are represented as \Rightarrow and \Leftarrow for the forward and backward direction, respectively. We define the reachability relationship as two nodes a and b in the graph model G of XML data satisfy reachability relationship if and only there is a path from a to b in G . Each edge in this path can be either edges representing nested relationship or reference relationship. We represent reachability by \rightsquigarrow . For example, $a \rightsquigarrow e$ is to retrieve all the e elements with a path from a to it. In Fig 1(b), this query will retrieve $d1, d2$ and $d3$.

The combination to reachability restraints may forms *subgraph query*. Subgraph query will retrieve the subgraphs of graph-structured XML matching the structure given by the query. The graph corresponding to the query is called *query graph*. The nodes in query graph represent the tag name of required elements. The edges in query graph represent the relationship between required elements. If an edge in query graph represents adjacent relationship, it is called *adjacent edge*. If an edge in query graph represents reachability relationship, it is called *reachability edge*. For an example, the query shown in fig 2(a) on XML document shown in fig 1(b) represents the query to retrieve all the subgraphs of it with structure a node connects to a c node, d node reaches to this c node and this c node reaches a f node. the result is shown in fig 6.

2.3 Reachability Coding

The goal of encoding XML is to represent the structural relationship so that the relationship between nodes in XML graph can be judged from the code quickly. With a good code, the query processing of structural query can be efficient. In this paper, we focus on *reachability coding*, which is used to judge the reachability relationship. We use an extension of reachability coding presented in [16]. In this coding, at first, all strongly connected components in the graph are contracted. Labeling is done by finding a spanning tree of the DAG generated in last step and assigning interval labels for nodes in the tree. The coding of the spanning tree is generated by post-order traversal. Each node is also assigned the number during traversal. The number is called *postid*. Next, to capture reachability relationships through non-spanning-forest edges, we add additional intervals to labels in reverse topological order of the DAG; specifically, if (u, v) is an edge not in the spanning forest, then all intervals of v are added to u (as well as labels of all nodes that can reach u). For an example, the reachability coding of graph in fig1(b) is shown in Fig 1(c). Using the spanning tree rooted at $a1$, we label $d2, f1$ with $[2, 2]$ and $[0, 0]$. In addition, $d2$ receives intervals from $f1$, resulting in that $b2$'s code is $[2, 2], [0, 0]$. In this coding, $a \rightsquigarrow b$ if and only if $b.postid$ is contained some interval associated with a .

3 Subgraph Join

In this section, we discuss the processing of subgraph queries. We present subgraph join algorithm and the method of preprocessing query and data to support subgraph join algorithm.

3.1 Preprocess of the Input

The interval labelling scheme of a graph is different from that of tree. There may be more than one intervals assigned to one node. The processing unit of our method is interval. So that we should assign the *postid* of each node to all of its intervals. If several intervals associated to nodes with the same tag have the same x and y value but different *postid*, they are merged. The result of this step is a list of intervals, each of which is associated with one or more *postids*. The list is called *candidate list*.

For the convenience of process, we will sort the intervals of all the nodes with the same tag by the value of x in ascending order and value of y in descending order. x is prior to y . It means only if two intervals have same x value, their y values are considered.

3.2 Preprocess for Subgraph Query

In order to apply subgraph join algorithm to process general subgraph query, some preprocess should be applied on the query when the query graph has circle.

If there are some circles in the query graph, a node n in each circle should be split to n_a and n_b break this circle. n_a includes all the incoming edges of n . n_b includes all the out edges of n . This node is the nodes related least edges in the circle.

When subgraph join is finished, the nodes in result corresponding to split query node are connected. Hash method is used.

Theorem 1. *After connection processing in the last step, the splitting of query node will not affect the final result of subgraph query.*

For the efficiency of query processing, before the process of data stated in Section 3.1, the nodes in the same SCC in each candidate list should be merged into one node. This node is called *stub node*. Since the coding of nodes in the same SCC have same intervals, the new node has these intervals, the number of the stub node is any of the number of the nodes belonging to the same SCC. Applying such preprocess is to prevent too large intermediate result during query processing without affecting the final result. For example, to process query shown in fig 3, there is a cycle in graph of the XML document with 100 a nodes, 100 b nodes and 100 c nodes respectively. Since they are reachable to each other, there will be 10^6 items in intermediate result after processing these nodes.

Corresponding to the merge, after the join is processed, the result should be extracted. The process of extraction is, for each result with stub node, from node set associated each merged nodes, one node is selected for one time to put on the position of the merged node. With a different combination of the selected nodes, one result is generated.

Theorem 2. *With extraction after all results are generated, the merging of nodes in the same SCC before query processing will not affect the final result.*

3.3 Data Structure for Subgraph Join

In our coding, there may be overlap in the intervals. Therefore, the stack based join of tree structured XML document can not be applied to our coding directly. We design a data structure, *interval stack*, to support efficient graph structural join. The interval stack is a DAG. Each node represents an interval. Each edge $e = n_1 \rightarrow n_2$ represents the interval of n_1 contains the interval of n_2 . The child of each node is sorted by the x values of the intervals.

There are two additional structures of the digraph, top and bottom. Top is the list of the sinks which are intervals without any interval containing them. Bottom is the list of sources which are intervals without any interval contained in them. They are both sorted by x of the intervals.

There are mainly two operators of interval stack, append and trim. The former is to append an interval to interval stack. The latter is to delete useless intervals from interval stack. During the performing of these two operations, the property of interval stack should be kept and top and bottom are maintained.

3.4 Subgraph Join Algorithms

With interval stack, we improve stack-based twig join [3] algorithm to support subgraph queries.

Of compacted interval list, we have following observations:

- The *postid* of a node is contained one and only one of its intervals.
- If two nodes have reachability relationship, it can and only can be checked by one interval. That is, if $a \rightsquigarrow b$, among all the intervals of the reachability of a , only one contain the $b.number$.

Suppose the input query can be visualized as a rooted DAG. The circle in input query will be broken in preprocess. If there is no root. A dummy root is added to the query.

The join candidates are a series lists of intervals with a list of nodes it corresponds to.

For each node in query graph, a structure is build which includes an interval stack(S) and its current cursor(C), the parents and children of it in query graph. The interval stack has the same function as that in structural join. M is a hash map, mapping *postid* of node to its children. The algorithms of subgraph join are described in Alg 1.

The subgraph join algorithm has two phases. In the first phase, each pair of nodes satisfying partial reachability relation described in query is outputted. In the second phase, the nodes in intermediate result unsatisfied the whole query are trimmed. Such nodes being included in intermediate result is because in the first phase, when each pair of nodes is outputted, only partial reachability relation related to these two node is considered. For an example, for query shown in fig 3, some of the intervals to process are shown in fig 4, the ids in brackets are the *postids* corresponding to the interval. Suppose the first number in bracket is in corresponding interval and others is not in the interval. During query processing, although a_{31} and c_{21} are not in final result, the pair (a_{31}, c_{21}) is still outputted.

During processing the query in fig 3 , interval a_1 contains interval c_1 . Based on observation 1, only pairs (a_{11}, c_{11}) , (a_{12}, c_{11}) , (a_{13}, c_{11}) are appended to intermediate result. This is because from the containment of these two intervals, only that c_{11} is in interval a_1 can be determined. So only the reachability of all nodes in the extent of a_1 and c_{11} is true.

getNext() is to find the next entry to process. It has similar function as *getNext* of twigjoin in [3]. First of all, the interval with least x value is chosen. If some intervals have same x value, the interval with largest y is chosen. If two intervals have same x and same y and their corresponding query nodes have reachability relation, the interval corresponding query node as ancestor is chosen. Otherwise, some result will be lost. For an example, consider query in fig 3. on the element sets visualized in fig 4, the interval a_1 has the same x and y as interval b_1 . The nodes corresponding to a_1 should be outputted with the nodes corresponding to b_1 and in the interval of b_1 . But if b_1 is chosen former than a_1 , these pairs will not be outputted. Since interval a_1 contains interval b_2 , the nodes corresponding to a_1 should be outputted with the nodes corresponding to b_2 and in the interval of b_2 . But if b_2 is chosen former, these pairs will lose.

Algorithm 1. GJoin(*root*)

```

1: while not end(root) do
2:   q = getNext(root)
3:   if not isSource(q) then
4:     if isSource(q) OR not emptyParent(q) then
5:       cleanNodes(q)
6:       push(q)
7:       advance(q)
8:   obtainResult()

1: function END(q)
2:   return  $\forall q_i : isSink(q_i) \Rightarrow end(q_i.C)$ 

1: procedure CLEARNODES(q)
2:   q.S.Trim(q.C)

1: function EMPTYPARENT(q)
2:   return  $\exists p_i \in q_i.parents : p_i.C = p_i.end$ 

1: procedure PUSH(q)
2:   for each node n  $\in q.C.context$  do
3:     if q = root then
4:       q.extent.add(n)
5:     if n.id > q.C.y then
6:       insertEntry(q.M, n)
7:       n.type = q
8:     else if n.id  $\geq q.C.x$  then
9:       for each p  $\in q.parents$  do
10:        pointTo(p,q,n.id)

1: procedure POINTTO(p,q,id)
2:   for each entry i  $\in p.S$  do
3:     if id  $\geq i.x$  AND id  $\leq i.y$  then
4:       for each node n  $\in i.context$  do
5:         M[n.id].child.add(id)

1: procedure OBTAINRESULT
2:   for each node n  $\in root.extent$  do
3:     b = generateResult(n)
4:     if b = FALSE then
5:       delete n from root.extent

1: function GENERATERESULT(node)
2:   if node is visited then
3:     return node.isresult
4:   b = TRUE
5:   for each child c of node do
6:     tb = generateResult(c)
7:     if tb = FALSE then
8:       delete c from node.child
9:       b = FALSE
10:    else if NOT c.type  $\in node.childtype$  then
11:      node.childtype.add(c.type)
12:    if node.childtype.size = node.type.child.size then
13:      node.isresult = TRUE
14:    return TRUE
15:  else
16:    node.isresult = FALSE
17:  return FALSE

```

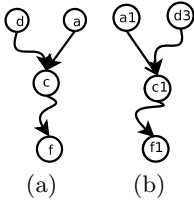
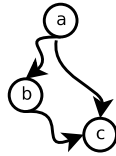
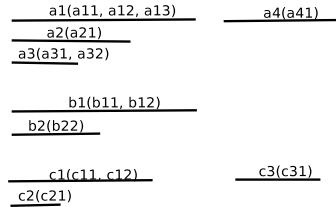
Algorithm 2. getNext(q)

```

1: function GETNEXT( $q$ )
2:   if isSink( $q$ ) then
3:     return  $q$ 
4:   for  $q_i \in q.children$  do
5:      $n_i = getNext(q_i)$ 
6:     if  $n_i.left < n_{min}.left$  then
7:        $n_{min} = n_i$ 
8:     else if  $n_i.left = n_{min}.left$  then
9:       if  $n_i.right > n_{min}.right$  then
10:         $n_{min} = n_i$ 
11:      else if  $n_i.right = n_{min}.right$  AND  $n_i$  is an ancestor of  $n_{min}$  then
12:         $n_{min} = n_i$ 
13:    $n_{max} = maxarg_{n_i} \{n_i.C.x\}$ 
14:   while  $q_i.C.y < q_{max}.C.y$  do
15:     advance( $q_i.C$ )
16:   if  $q_i.C.x \leq q_{min}.C.x$  AND  $q_i.C.y \geq q_{min}.C.y$  then
17:     return  $q$ 
18:   else
19:     return  $n_{min}$ 

```

Note the function *emptyParent()* is to check whether the nodes in current interval satisfies the restriction of all incoming paths in the query. In our example, when interval c_3 is met, since interval stack of b is empty, it will not be considered.

**Fig. 2.** Example Queries**Fig. 3.** Example Query**Fig. 4.** Element sets for fig 3

Outputted pairs are organized by the ancestors. The main memory may be not enough to store intermediate results. External memory is used to store intermediate results. Since each node may have more than one descendant during query processing, children of one node are stored as a list in disk. The head of the list associated with a node record the number of the node, the query node corresponding to the node and the pointer to the first entry of the list. Each of entries in the list includes a 2-ary, $(node, next)$, where *node* is the pointer to the node this entry corresponding to and *next* is the pointer to next entry of the list. In the hash map, each entry e_n corresponds one node n . Each entry contain the head of the the position of the head and tail of list of n .

Theorem 3. *The logical I/O number of subgraph join algorithms is linear to the number to the pair of nodes satisfying the reachability relationship described in query.*

4 Experiments

In this section, we present results and analysis of part of our extensive experiment of subgraph join algorithms based on reachability coding.

4.1 Experimental Setup

The Testbed. All our experiments were performed on a PC with Pentium 1GMHZ CPU, 256M main memory and 30G IDE hard disk. The OS is Windows 2000 Professional. We implemented all the algorithms using Microsoft Visual C++ 6.0. We implemented the encoding of graph and subgraph join algorithms. We use LRU policy for buffer replacement.

For comparison, we also implemented F&B index [13] for graph structured XML document. F&B index supports all the subgraph queries for XML.

Dataset. The dataset we tested is the standard XMark benchmark dataset[21]. We used scale factor 0.1, 0.2, 0.3, 0.4 and 0.5, which generated XML document with size 10M, 20M, 30M, 40M and 50M respectively. It has complicated schema, including circle.

Some statistics information of test XML documents are shown in Table 1.

Table 1. Information of Test Document

Document size	11.3M	22.8M	34.0M	45.3M	56.2M
Node number	175382	351241	524067	697342	870628
Edge number	206129	413110	616228	820437	1024072

Query Set. In order to better test and understand the characteristics of the algorithms, we designed a set of queries that has different characteristics. We design three queries. They represent various structures. The query graph of them are shown in fig 5(a), fig 5(b) and fig 5(c), respectively.

4.2 Changing System Parameters

In this subsection, we investigate the performance of our system by varying various system parameters. We use physical I/O and run time to reflect the impact of different parameter setting.

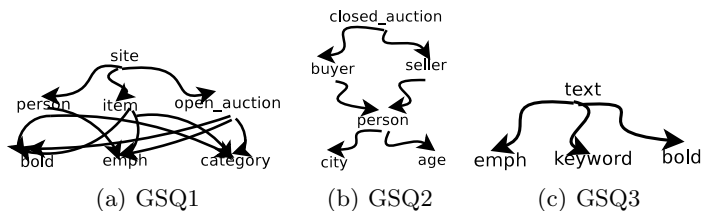


Fig. 5. Test Queries

Scalability Experiment. We test the queries on XML documents with various sizes. In order to test the scalability of the subgraph join algorithm. We choose SGQ2 and SGQ3 as test query. We fix main memory 8M and block size 4096. The results are shown in fig 6(a) and fig 6(b), respectively. SGQ1 is a simple twig query. The nodes related to SGQ1 in XML document is not in any SCC and all have single parent. Therefore, the increase trend is nearly linear. SGQ2 is a complex subgraph query. One person node may be reached by more than one seller nodes and only parts of person nodes are reached by both seller node and buyer node. The trend of run time is faster than linear but still slower than square.

Varying Buffer Size. The physicalIO change with block number of SGQ1 is shown in fig 6(c). From the fig 6(c), we can find that without enough main memory, the second phase result more physical I/O than the first phase. This is because in the second phase the whole intermediate result is traversed while in the first phase, the operation is mainly append.

4.3 Comparison Experiment

We do comparison in 10M XML document. Its F&B-index has 167072 nodes. We naive implemented the depth first traversal-based query processing by F&B-index. The reason why we do not compare larger XML document is that when XML document gets larger, the query processing in F&B-index becomes too slow.

The result of comparison subgraph query process efficiencies of subgraph join algorithm and F&B index is shown in Fig 6(d). Y axis is in log scale. subgraph join algorithm outperforms the efficiency of F&B index. For SGQ1, the efficiency are similar. It is because the nodes in XML document related to SGQ1 is in tree structured in Xmark document and the search depth in F&B index is limited.

5 Related Work

With efficient coding, XML queries can also be evaluated on-the-fly using the join-based approaches. Structural join and twig join are such operators and their efficient evaluation algorithms have been extensively studied [27,14,8,10,5,25] [3,11]. Their basic tool is the coding schemes that enable efficient checking of

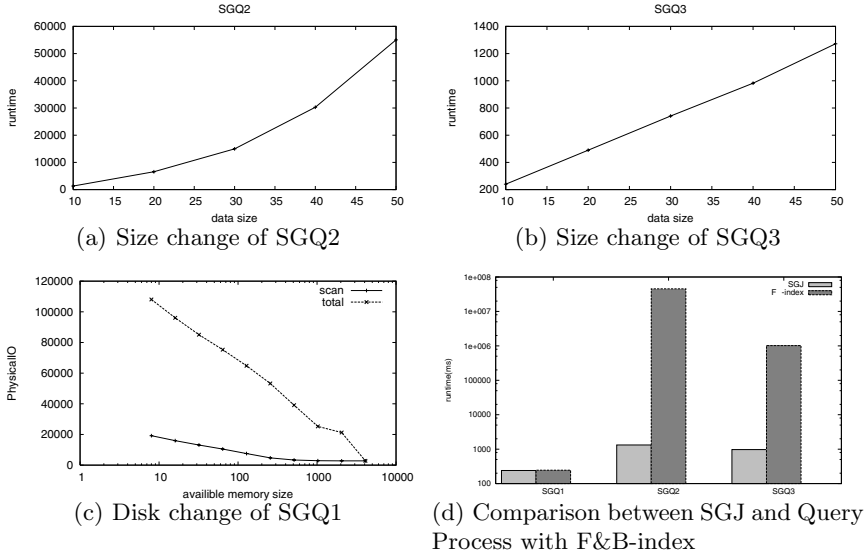


Fig. 6. Experiment Results

structural relationship of any two nodes. TwigStack [3] is the best twig join algorithm to answer all twig queries without using additional index. The idea of these work can be referenced to process query on graph. But these algorithms can not be applied on the coding of graph directly.

6 Conclusions

Information in some applications can be naturally stored as graph modeled data. The processing of graph structured XML data brings new challenges. To process structural query on graph structured XML data, in this paper, we present reachability labelling scheme for graph structured XML. With such labelling scheme, the reachability relationship between two nodes in graph structured XML can be judged efficiently. Based on the labelling scheme, we design graph structural join and subgraph join algorithms of graph structured XML to perform subgraph queries. From experiment, our labelling scheme has acceptable size. The subgraph join algorithm outperforms the query processing with F&B-index.

Our further work includes designing efficient index structure so support efficient query processing on graph structured XML document.

References

1. *Introduction to Algorithms*. MIT Press, Cambridge MA, 1990.
2. Shurug Al-Khalifa, H. V. Jagadish, Jignesh M. Patel, Yuqing Wu, Nick Koudas, and Divesh Srivastava. Structural joins: A primitive for efficient XML query pattern matching. In *Proceedings of the 18th International Conference on Data Engineering (ICDE 2002)*, pages 141–152, 2002.

3. Nicolas Bruno, Nick Koudas, and Divesh Srivastava. Holistic twig joins: Optimal XML pattern matching. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 2002)*, pages 310–321, 2002.
4. Donald D. Chamberlin, Daniela Florescu, and Jonathan Robie. XQuery: A query language for XML. In *W3C Working Draft*, <http://www.w3.org/TR/xquery>, 2001.
5. Shu-Yao Chien, Zografoula Vagena, Donghui Zhang, Vassilis J. Tsotras, and Carlo Zaniolo. Efficient structural joins on indexed XML documents. In *Proceedings of 28th International Conference on Very Large Data Bases (VLDB 2002)*, pages 263–274, 2002.
6. James Clark and Steve DeRose. XML path language (XPath). In *W3C Recommendation, 16 November 1999*, <http://www.w3.org/TR/xpath>, 1999.
7. Haim Kaplan Uri Zwick Edith Cohen, Eran Halperin. Reachability and distance queries via 2-hop labels. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '02)*, pages 937–946, San Francisco, CA, USA, January 2002.
8. Torsten Grust. Accelerating XPath location steps. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 2002)*, pages 109–120, Hong Kong, China, August 2002.
9. Ronen Shabo Haim Kaplan, Tova Milo. A comparison of labeling schemes for ancestor queries. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '02)*, pages 954 – 963, San Francisco, CA, USA, January 2002.
10. Haifeng Jiang, Hongjun Lu, Wei Wang, and Beng Chin Ooi. XR-Tree: Indexing XML data for efficient structural join. In *Proceedings of the 19th International Conference on Data Engineering (ICDE 2003)*, pages 253–263, 2003.
11. Haifeng Jiang, Wei Wang, Hongjun Lu, and Jeffrey Xu Yu. Holistic twig joins on indexed xml documents. In *Proceedings of 29th International Conference on Very Large Data Bases (VLDB 2003)*, pages 273–284, 2003.
12. Tiko Kameda. On the vector representation of the reachability in planar directed graphs. *Information Process Letters*, 3(3):78–80, 1975.
13. Raghav Kaushik, Philip Bohannon, Jeffrey F. Naughton, and Henry F. Korth. Covering indexes for branching path queries. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 2002)*, pages 133–144, 2002.
14. Quanzhong Li and Bongki Moon. Indexing and querying XML data for regular path expressions. In *Proceedings of 27th International Conference on Very Large Data Base (VLDB 2001)*, pages 361–370, 2001.
15. Tova Milo and Dan Suciu. Index structures for path expressions. In *Proceedings of the 7th International Conference on Database Theory (ICDE 1999)*, pages 277–295, 1999.
16. H. V. Jagadish Rakesh Agrawal, Alexander Borgida. Efficient management of transitive relationships in large data and knowledge bases. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data (SIGMOD 1989)*, pages 253–262, Portland, Oregon, May 1989.
17. Gerhard Weikum Ralf Schenkel, Anja Theobald. Hopi: An efficient connection index for complex xml document collections. In *Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology (EDBT04)*, pages 237–255, Heraklion, Crete, Greece, March 14-18 2004.
18. Ioannis G. Tollis Roberto Tamassia. Dynamic reachability in planar digraphs with one source and one sink. *Theoretical Computer Science*, 119(2):331–343, 1993.

19. A. Sayed and R. Unland. Indexing and querying heterogeneous xml collections. In *Proceedings of In 14th International Conference on Computer Theory and Applications*, Alex, Egypt, September 2004.
20. Ralf Schenkel. Flix: A flexible framework for indexing complex xml document collections. In *Proceedings of International Workshop on Database Technologies for Handling XML Information on the Web(DATAAX04)*, Heraklion, Crete, Greece, March 2004.
21. Albrecht Schmidt, Florian Waas, Martin L. Kersten, Michael J. Carey, Ioana Manolescu, and Ralph Busse. XMark: A benchmark for XML data management. In *Proceedings of 28th International Conference on Very Large Data Bases (VLDB 2002)*, pages 974–985, 2002.
22. Theis Rauhe Stephen Alstrup. Small induced-universal graphs and compact implicit graph representations. In *Proceedings of 2002 IEEE Symposium on Foundations of Computer Science (FOCS '02)*, pages 53–62, Vancouver, BC, Canada, November 2002.
23. C. M. Sperberg-McQueen Francois Yergeau Tim Bray, Jean Paoli. Extensible markup language (xml) 1.0 (third edition). In *W3C Recommendation 04 February 2004*, <http://www.w3.org/TR/REC-xml/>, 2004.
24. Michel Scholl Sotirios Tourtounis Vassilis Christophides, Dimitris Plexousakis. On labeling schemes for the semantic web. In *Proceedings of the Twelfth International World Wide Web Conference(WWW2003)*, pages 544–555, Budapest, Hungary, May 2003.
25. Wei Wang, Haifeng Jiang, Hongjun Lu, and Jeffrey Xu Yu. PBiTree coding and efficient processing of containment joins. In *Proceedings of the 19th International Conference on Data Engineering (ICDE 2003)*, pages 391–402, 2003.
26. Joseph Gil Yoav Zibin. Efficient subtyping tests with pq-encoding. In *Proceedings of the 2001 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA 2001)*, pages 96–107, San Francisco, CA, USA, October 2001.
27. Chun Zhang, Jeffrey F. Naughton, David J. DeWitt, Qiong Luo, and Guy M. Lohman. On supporting containment queries in relational database management systems. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD 2001)*, pages 425–436, 2001.
28. Vassilis J. Tsotras Zografoula Vagena, Mirella Moura Moro. Twig query processing over graph-structured xml data. In *Proceedings of the Seventh International Workshop on the Web and Databases(WebDB 2004)*, pages 43–48, 2004.

A Soft Real-Time Web News Classification System with Double Control Loops

Huayong Wang, Yu Chen, and Yiqi Dai

Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, P.R.China
wanghy02@mails.tsinghua.edu.cn, yuchen@tsinghua.edu.cn,
dyq@theory.tsinghua.edu.cn

Abstract. This paper proposes a framework for soft real-time text classification system, which use control theory as a scientific underpinning, rather than ad hoc solutions. In order to provide real-time guarantee, two control loops are adopted. The feed forward control loop estimates the suitable number of classifiers according to the current workload, while the feedback control loop provides fine-grained control to the number of classifiers that perform imprecise computation. The soft real-time classification system can accommodate to the change of workload and transitional overload. The theory analysis and experiments result further prove its effectiveness: the variation range of the average response time is kept within $\pm 3\%$ of the desired value; the computational resource is dynamically reallocated and reclaimed.

1 Introduction

The amount of online text data has grown greatly in recent years because of the increase in popularity of the World Wide Web. As a result, there is a need to provide efficient content-based retrieval, search, and filtering for these huge and unstructured online repositories. One of the major applications in this aspect is search engine, which fetches, classifies and indexes the web information. Usually, a full-fledged search engine includes a front-end search subsystem and a back-end data processing subsystem. The front-end follows the html links and fetches the web pages, while the back-end is in responsible for classification and indexing. In this paper, we consider the problem of how to design a soft real-time classification subsystem for a web news search system. The bulk of the research on text classification (TC) system has focused on improvements to "precision" or "recall", and is not concerned with how long it takes to complete the tasks. Our contribution is to take the time constraints into consideration, and propose a general framework to design a text classification system with soft real-time guarantee.

Real-time systems are computing systems that must react within precise time constraints to events in the environment. As a consequence, the correct behavior of these systems depends not only on the value of the computation but also on the time at which the results are produced. The time before which a task should be completed to avoid damage to the system is called deadline. Real-time system should try its best to guar-

antee all running tasks to finish before their respective deadlines. A system is said to be soft real-time if deadline missing decreases the Qos of the system, but does not jeopardize the system's correct behavior. With the rapid growth of textual information available on the Internet, the search engine front-end is already able to fetch more and more web pages in a given time. In order to keep up with the high-speed of the front-end, it is most desirable that the classification subsystem should provide soft real-time service.

However, very limited research work has been done in this direction, and the evaluation to the existing systems is typically conducted experimentally, rather than analytically. The reason is that, in order to evaluate a system analytically, we would need a formal mathematical model for the system, and the classification system is such a nonlinear and time-varying system that is difficult to be formalized inherently. Without an analyzable mathematical model, it is impossible to apply real-time theory to the system design. How to provide real-time guarantee without relying on over-provisioning of system resource becomes a great challenge.

After this introduction, section two introduces the related work. Section three explains our system design method in theory. Section four gives the experiments result. Section five makes the conclusion.

2 Related Work

Since the design of real-time TC system requires the knowledge of information retrieval, real-time system and control theory. This section introduces the related work in all these fields.

Current TC algorithms include Naive Bayes, k-Nearest Neighbor (KNN), Linear Least Squares Fit (LLSF), Neural Network, Boosting, Support Vector Machine (SVM) and so on [1]. Based on these algorithms, great effort has been made by research community to reduce the computational complexity and hence increase the running speed. Yang provides a formal analysis of the computational complexities of five popular TC algorithms [2]. Grossman introduce some techniques to improve the runtime performance of information retrieval system with inverted index and signature files [3]. The book also discusses the system implementation on parallel processors. Paper [4,5,6] presents various methods to reduce the runtime complexity, for example, density-based method, pruning method, and discriminating power method. However, all these methods rely on powerful hardware or algorithm complexity reduction. If the test document set is a static set, the results of aforementioned methods are acceptable. While facing the large amount of textual information from Internet, the requirement is quite different. How to integrate these classification methods into a practical real-time system is not investigated by previous researches.

Real-time Task Model is the core concept of real-time scheduling systems. The most famous scheduling algorithms for real-time tasks are RM and EDF, which is firstly proposed by Liu and Layland in 1973 [7] and has been thoroughly studied [8,9]. In these researches, a real-time system is modeled as a set of independent tasks.

$\tau = \{\tau_1, \dots, \tau_n\}$ denotes a task set containing n tasks. A real-time task τ_i is characterized by the following parameters: 1) Arrival time A_i is the time at which a task becomes ready for execution. 2) Start time S_i ($S_i \geq A_i$) is the time at which a task starts

its execution. 3) Finishing time F_i ($F_i > S_i$) is the time at which a task finishes its execution. 4) Deadline D_i is the time before which a task should be completed to avoid damage to the system. Obviously, " $F_i > D_i$ " means deadline missing. 5) Relative deadline L_i is equal to $D_i - A_i$. An important metric in this paper to evaluate real-time performance is average response time:

$$\bar{R} = \frac{1}{n} \sum_{i=1}^n (F_i - A_i) . \quad (1)$$

Average response time consistently longer than the desired relative deadline is unacceptable to the users. Average response time consistently shorter indicates over-provisioning of resource that could have been used for other users and applications.

Imprecise computation is a useful technique in real-time system. In an imprecise computation system, every task is decomposed to a mandatory subtask and an optional subtask. The mandatory subtask is the portion of the computation that must be done in order to produce a result of acceptable quality, whereas the optional subtask refines this result [10]. Web news classification system is suitable for imprecise computation model for its characteristics of hierarchical classification. Hierarchical classification refers to assigning of one or more suitable categories from a hierarchical category space to a document. Web news classification system adopts a top-down level-based method that can classify news to both leaf and internal categories in the category tree. Our hierarchical category space contains ten major categories, and each major category contains several minor categories. For example, "sport" is a major category while "football" and "basketball" are its minor categories. If the system is overload, it can omit the minor categories and only classifies news according to major categories. The result is trading off computation accuracy with temporal requirements. Imprecise computation model dose not change the essence of the classification algorithms, so most of the current TC algorithms are able to support this model inherently. The real-time framework in this paper dose not target for one specific classification algorithm. Choosing what classification algorithm depends on the system implementers.

Feedback control is a powerful concept that has played a pivotal role in the development of many areas of engineering. It has several advantages. Using feedback, it is possible to create linear behavior out of non-linear components and to modify the dynamic behavior of a system. The advantage of using feedback in conjunction with real-time scheduling is that precise real-time task model can be relaxed [11]. This feature is especially desirable for TC system, because TC system runs in an unpredictable environment, where the execution time of a task varies greatly from one instance to another. However, feedback control theory has been mostly applied in mechanical and electrical systems. In trying to apply feedback control theory to a TC system, the modeling and implementation face significant research challenges. This paper tries to answer some of these challenges.

3 System Components

The dominant approach adopted by TC systems is based on machine learning techniques: a general inductive process automatically builds a classifier by learning, from a

set of pre-classified documents, the characteristics of the categories [1]. A classifier is a program devoted to the classification work. We can speedup the TC system by running multiple classifiers. However, it is generally believed that more classifiers will consume more computing resources, which is undesirable. So, we now raise two questions. 1) How many classifiers are suitable for the current workload? 2) How to accurately develop a linear feedback model to overcome the transient overload due to stochastic environment? These two questions lead to the following system components.

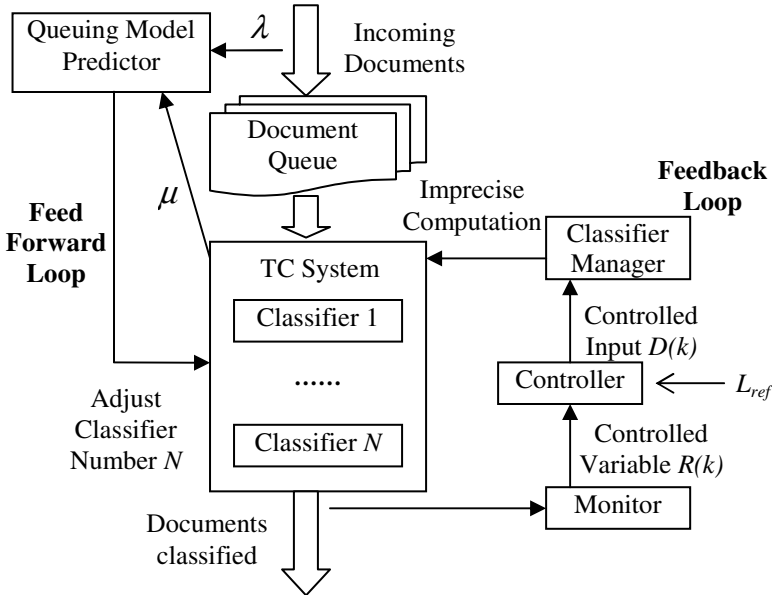


Fig. 1. Framework of real-time text classification system

As shown in Fig. 1, the incoming documents are put into a document queue. The system assigns a desired relative deadline L_{ref} for all documents in the queue. From the user's viewpoint, L_{ref} is the maximum allowable delay between the document's arrival and leaving. In order to provide timely service, the system contains two control loops: feed forward control loop and feedback control loop. In the feed forward control loop, classical results from queuing theory are used for computing the number of classifiers necessary to achieve the specified deadline requirement given the current observed average document arrival rate λ . In the feedback control loop, the average response time $R(k)$ is observed by the Monitor, and compared to L_{ref} by the Controller. And then, Classifier Manager adjusts those classifiers according to the imprecise computation model. The feed forward and feedback components operate concurrently in a complementary manner.

3.1 Feed Forward Loop

The feed forward loop is relatively simple and easy to design. Its central idea is using an $M/M/N$ queuing model [12,13]. Let the document arrival process is a Poisson process

with arrival rate λ , and the average service rate of one classifier is μ . According to $M/M/N$ queuing model, the average delay of a document in the system is

$$W = \frac{\rho^N P_0}{N\mu \times N!(1 - \frac{\rho}{N})^2} + \frac{1}{\mu}, \quad (2)$$

where $\rho = \lambda / \mu$ and P_0 is

$$P_0 = \left[\left(\sum_{i=0}^N \frac{\rho^i}{i!} \right) + \frac{\rho^{N+1}}{N!(N - \rho)} \right]^{-1}. \quad (3)$$

If λ and μ are known, we can compute the lower bound of classifier number N by

$$W \leq L_{ref}. \quad (4)$$

In Fig. 1, queuing model predictor completes the above computation work. There are three points to be explained for the queuing model predictor:

- 1) The average arrival rate λ is monitored by queuing model predictor over an observation window of 500 incoming documents. λ may change abruptly since the strength of the workload is variable.
- 2) Compared to parameter λ , average service rate μ is relatively stable because the classification algorithm is fixed. The actual value of μ is measured from the actual system by recording how many documents are classified by one classifier on average over an observation window.
- 3) Queuing model predictor adjusts the classifier number according to formula 4. If the parameter μ is relatively constant, the numerical relationship between N and λ can be pre-computed offline. Therefore the computational complexity during runtime is quite small. In the following discussion, the parameter μ and λ are assumed to be known.

The feed forward loop can figure out the suitable classifier number by queuing model. However, that is not enough for the real-time guarantee because the transitional change of the workload cannot be observed by queuing model predictor. The feedback control loop is designed to resolve this problem.

3.2 Feedback Loop

The feedback loop adjusts the system in an incremental manner. First of all, it is necessary to decide the following variables in terms of control theory. Readers can refer to [14,15,16] for the basic knowledge of control theory.

Controlled variable represents the performance metric controlled by the real-time TC system. In our model, the variable "average response time" $R(k)$ is used as controlled variable, which is defined over a time window $\{(k-1)W, kW\}$, where W is the sampling period and k is called the sampling instant.

Performance reference is the desired relative deadline L_{ref} of the system. The difference between L_{ref} and the current value of the controlled variable is called an error. The "average response time error" $E(k) = L_{ref} - R(k)$.

Manipulated variable is the system attribute that can be dynamically changed to affect the value of the controlled variable. In our system, the manipulated variable is "classification speed" $S(k)$. The classification speed $S(k)$ at the k th sampling instant is defined as

$$S(k) = (N - M(k)) \times \mu + M(k) \times (\mu + \Delta) , \quad (5)$$

where μ is the average service rate of one classifier, $\mu + \Delta$ is the average service rate of one classifier that performs imprecise computation, N is the current number of all classifiers, $M(k)$ is the current number of classifiers that perform imprecise computation. The classifiers performing imprecise computation throw away the optional part of the task, so their service rate is higher than the normal classifiers. $\mu + \Delta$ can also be measured from the actual system like μ .

Fig. 1 illustrates the feedback control loop. The monitor measures the controlled variable $R(k)$ and feeds the samples back to the controller. The controller compares the performance references L_{ref} with $R(k)$ to get the current error $E(k)$, and computes a change $D(k)$ (called control input) to the speed $S(k)$. The classifier manager dynamically changes the classification speed at each sampling instant k according to the control input $D(k)$ by adjusting the number of classifiers that perform imprecise computation. The goal of the classifier manager is to enforce the new speed $S(k+1) = S(k) + D(k)$. According to formula 5, the number of the classifiers that perform imprecise computation in next time window can be calculated by

$$M(k+1) = \max \left\{ 0, \left\lceil \frac{S(k+1) - N \times \mu}{\Delta} \right\rceil \right\} . \quad (6)$$

With the aforementioned definitions, it is possible to establish an analyzable model to approximate the feedback loop in the TC system. Although it is difficult to precisely model a nonlinear and time-varying system, we can approximate such a system with a linear model for the purpose of control design because of the robustness of feedback control with regard to system variations. Starting from the control input, the classifier manager changes the classification speed at every sampling instant:

$$S(k+1) = S(k) + D(k) . \quad (7)$$

$R(k)$ usually decreases nonlinearly with the classification speed $S(k)$. The relationship between $R(k)$ and $S(k)$ needs to be linearized by taking the derivative at the vicinity of the performance reference L_{ref} , as the "response time factor" R_N :

$$R_N = \left. \frac{dR(k)}{dS(k)} \right|_{R(k)=L_{ref}} . \quad (8)$$

In practice, the response time factor R_N can be estimated experimentally by plotting a $R(k)$ curve as a function of $S(k)$ based on experimental data if the classifier number N is

known. Given the response time factor, we have the following linearized formula for the purpose of control design,

$$R(k) = R(k-1) + R_N(S(k) - S(k-1)) \quad (9)$$

Based on Equations (7) and (9), the analyzable model for the average response time output is as

$$R(k) = R(k-1) + R_N D(k-1) . \quad (10)$$

We now convert this model to z-domain transfer function that is suitable to control theory analysis. Let $Y(z)$ be the z-transform of the output variable $R(k)$, and $X(z)$ be the z-transform of the input variable $D(k)$. A linear system can be represented by a transfer function $H(z) = Y(z)/X(z)$. For our system, formula (10) can be converted into z-domain as

$$Y(z) = z^{-1}Y(z) + R_N X(z)z^{-1} , \quad (11)$$

so the transfer function in open loop is

$$H_{open}(z) = \frac{R_N}{z-1} . \quad (12)$$

At each sampling instant k , the controller computes a control input $D(k)$ based on the average response time error $E(k)$. We choose a simple P (proportional) control function to compute the control input:

$$D(k) = KE(k) . \quad (13)$$

It is obvious that the system performance can be adjusted by parameter K . System implementors can choose suitable value of K for their own systems.

4 Experiment Results

The hardware platform of our experiments is a cluster of four PCs with CPU800MHz and memory 256M. In a single CPU system, when the number of classifier increases, the efficiency cannot increase linearly. The reason is that those classifiers run concurrently, rather than parallelly. In our experiment, multiple computers work together to run multiple classifiers in order to alleviate the hardware bottleneck of the single CPU system. This test system adopts KNN algorithm (the system does not depend on one specific classification algorithm), and the category space contains ten major categories. The front-end search subsystem provides about 15000 web pages per hour to the TC subsystem on average. The size of these web pages is about 25k bytes on average, including the HTML tags. If the HTML tags are filtered out, the size of the text inside each web page is about 2k-5k. In this test system, L_{ref} chooses the value of 5 minutes. Fig. 2 records the relationship between average service rate μ and classifier number N . The values presented in Fig. 2 are average values of hundreds of tests.

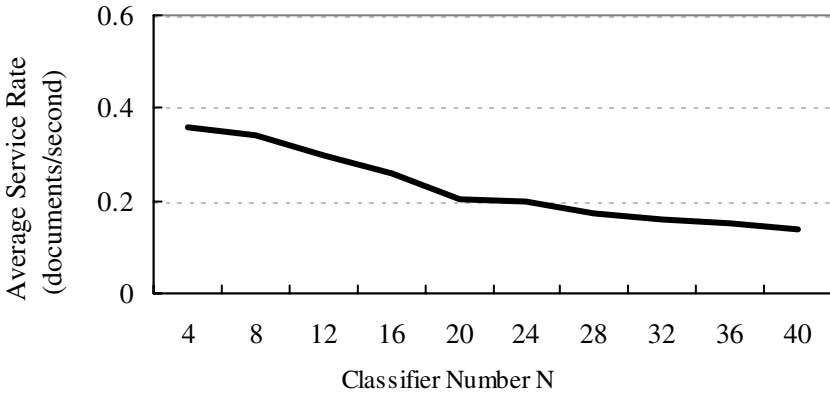


Fig. 2. The relationship between average service rate and classifier number

The next step is to determine the relationship between $R(k)$ and $S(k)$ with a given value of N . Fig. 3 shows the experiment results of $R(k)$ based on $S(k)$ when N equals to 8, 12, and 16. The relationship when N equals to other values is not drawn here because of the space limit. If $L_{ref} = 5$ minutes, we can draw the tangent line at point $R(k) = 5$, the tangent slope is

$$R_N = \left. \frac{dR(k)}{dS(k)} \right|_{R(k)=L_{ref}} = \begin{cases} -0.1 & N = 8 \\ -0.02 & N = 12 \\ -0.01 & N = 16 \\ \dots & N = other \end{cases} \quad (14)$$

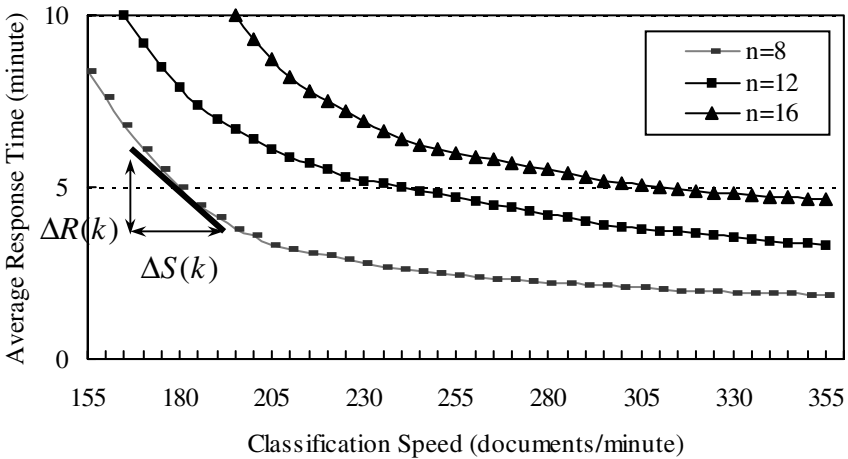


Fig. 3. $R(k)$ curve based on $S(k)$

Figure 4 records the system performance in 100 successive time windows during stable state. Each time window is 2 minutes. While the disturbance exists, the variation range of $R(k)$ is within $\pm 3\%$ to L_{ref} .

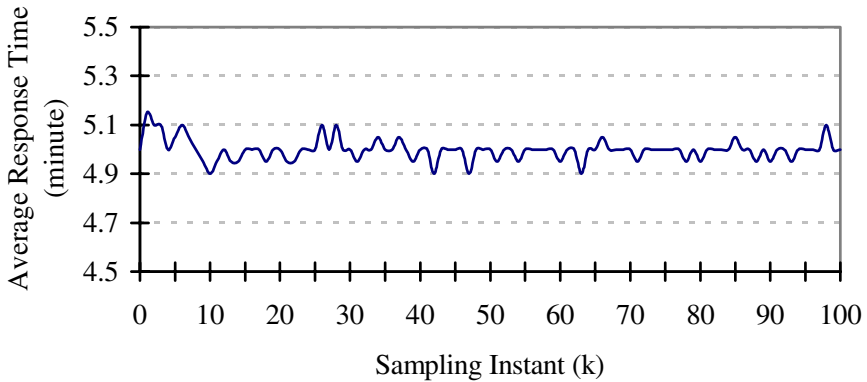


Fig. 4. Stable state performance

Fig. 5 records the change of classifier number during the same time segment. With the soft real-time guarantee, the number of the classifiers are dynamically increased or decreased to save the computation resource, which is one of the advantages of our system. Since $R(s)$ is stabilized at the point of L_{ref} , the soft real-time guarantee is obtained without relying on over-provisioning of system resource.

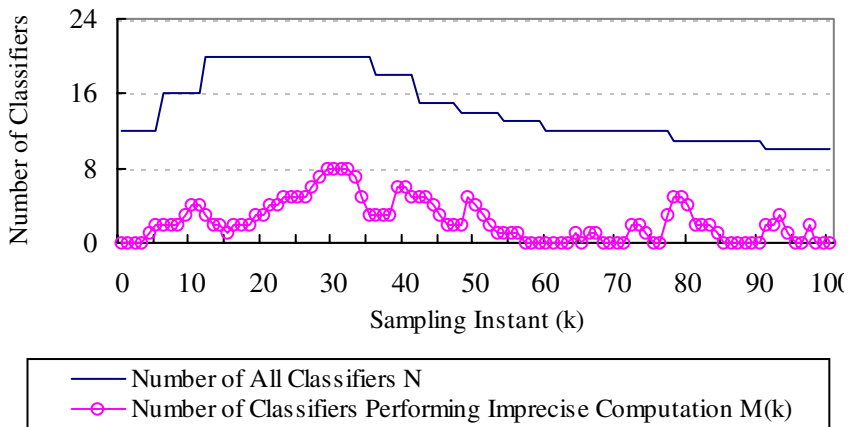


Fig. 5. Chang of classifier number

5 Conclusions

This paper proposes a design method for soft real-time TC system. A mathematical model of TC system is established. And the control theory is applied to prove the system performance guarantee. Experiment results further prove the effectiveness of our scheme. Future work will further investigate the feedback control of classifiers and integrate the multiple different classification algorithms into one real-time framework.

References

1. Sebastiani, F.: Machine learning in automated text categorization. *ACM Computing Surveys*. 34(1) (2002) 1-47
2. Yang, Y., Zhang, J., Kisiel, B.: A scalability analysis of classifiers in text categorization. Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. ACM Press, New York (2003) 96-103
3. Grossman, D.A., Frieder, O.: *Information Retrieval - Algorithms and Heuristics*. Kluwer Academic Publishers, Massachusetts (1998)
4. Li, R.-L., Hu, Y.-F.: Noise reduction to text categorization based on density for KNN. Proceedings of the International Conference on Machine Learning and Cybernetics, Vol. 5. IEEE Computer Society, California (2003) 3119-3124
5. Zhou, S., Ling, T.-W., Guan, J., et al.: Fast text classification: a training-corpus pruning based approach. Proceedings of 8th International Conference on Database Systems for Advanced Applications. IEEE Computer Society, California (2003) 127-136
6. Deng, Z.-H., Tang S.-W., Yang D.-Q., et al.: SRFW: a simple, fast and effective text classification algorithm. Proceedings of International Conference on Machine Learning and Cybernetics, Vol. 3. IEEE Computer Society, California (2002) 1267-1271
7. Liu, C.L., Layland, J.W.: Scheduling algorithms for multiprogramming in a hard real time environment. *Journal of the ACM*, 20 (1973)
8. Krishna, C.M., Shin, K.G.: *Real-Time Systems*. McGraw-Hill Companies, Columbus (1997)
9. Buttazzo, G.C.: Rate monotonic vs. EDF: judgment day. *Real-Time Systems*. 28(1) (2005) 5-26
10. Buttazzo, G.C.: *Hard Real-Time Computing System: Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers, Massachusetts (2000)
11. Sha, L., Abdelzaher, T.F., Arzen K. -E., et al.: Real time scheduling theory: a historical perspective. *Real-Time Systems*. 28(2/3) (2004) 101 - 155
12. Lu, F.S.: *Queueing Theory and Its Applications*. Hunan Scientific Publication Press, Hunan, P.R.China (1984)
13. Lin, Y.L.: *Applied Stochastic Process*. Tsinghua University Press, Beijing, P.R.China (2002)
14. Cheng, S., Guo, B., Li, X., et al.: *Signal and System*. Press of Xian Electronic Technology University, Xian, P.R.China (2001)
15. Shun, Z.: *System Analysis and Control*. Tsinghua University Press, Beijing, P.R.China (1994)
16. Franklin, G.F., Powell, J.D., Workman, M.L.: *Digital Control of Dynamic Systems*. 3rd edn. Addison-Wesley, Boston (1998)

Searching the World Wide Web for Local Services and Facilities: A Review on the Patterns of Location-Based Queries

Saeid Asadi¹, Chung-Yi Chang¹, Xiaofang Zhou¹, and Joachim Diederich^{1,2}

¹ School of ITEE, The University of Queensland, Brisbane, QLD 4072, Australia
{asadi, julie, zxf, joachimd}@itee.uq.edu.au

² Department of Computer Science, School of Engineering, American University of Sharjah,
P.O. Box: 26666, Sharjah, UAE
jdiederich@aus.edu

Abstract. Many queries sent to search engines refer to specific locations in the world. Location-based queries try to find local services and facilities around the user's environment or in a particular area. This paper reviews the specifications of geospatial queries and discusses the similarities and differences between location-based queries and other queries. We introduce nine patterns for location-based queries containing either a service name alone or a service name accompanied by a location name. Our survey indicates that at least 22% of the Web queries have a geospatial dimension and most of these can be considered as location-based queries. We propose that location-based queries should be treated different from general queries to produce more relevant results.

1 Introduction

Many search engines have been designed to capture webpages and other resources on the Web. On Google alone, 250 to 300 million queries are run every day [1]. People search the Web for many reasons and this is reflected in the search queries. Spink et al. [2] report some changes in Web search topics from entertainment and sex in 1997 to commerce, travel, employment, economy and finding people and places in 2001. This indicates that people are gradually increasing use of the Web in responding their life's requirements. Although a certain part of Web queries contain location names or have geospatial dimensions such as driving, shopping or accommodation, general search engines do not provide results in a useful way based on the requested locations; and therefore, users cannot easily find the webpages related to the geographical locations [3].

This paper sets out to address challenges of location-based search through analysing location-based queries. We first review previous work on geospatial information retrieval in the context of the Web and then offer several patterns for location-based queries. We believe that by distinguishing location-based query patterns, search engines would be able to develop more effective location-based services.

2 Background

Spatial data as has been described in GIS Glossary [4] “is the data pertaining to the location of geographical entities together with their spatial dimensions”. Spatial data is also known as geospatial data or geographic information [5] and has been studied for years. However, there are many issues with geospatial information retrieval on the Web which affect search engine results among them the ambiguity of location names and multiple locations related to a webpage are notable.

Some basics concepts are adopted from other fields especially geography. In 1984, basic themes of geography [6] were defined to facilitate the education of geography. The five themes of geography are movement, human-environment interaction, location, place and region. Based on these themes, location refers to “where” something is while place says “what” it is. Location could be absolute or relative. Absolute location is shown by an address. Relative location indicates an address by referring to its relation with other places. Relative location is useful when we have a reference point. For example, we can say the British Museum is located in the centre of London 3 kilometres from City University.

In the context of the Web, location can be defined as “a place on the Internet where an Internet resource, such as a webpage, is stored” [7]. However, in many cases, this is not a sufficient definition. So far, efforts for making a world-wide location-based search engine have been unsuccessful. Current search engines have offered some location-aware facilities. Google Local¹ can search for local business information in the USA and Canada [8]. Yahoo Local² has a similar function for the USA. The SPIRITS project has tried to make a universal location-based search tool through combining geographical ontologies with query expansion, machine learning and geospatial ranking [9]. Other geographical search tools such as Northern Light³ or GeoSearch⁴ offer limited services. Localization of a website is the procedure of designing a website based on local cultures and needs, and it consists of three levels: identification of website subjects, identification of the target culture and applying localized aspects in the design process [10]. Major general search engines such as Yahoo, Google and MSN have developed local services for different countries, regions and languages. Using mobile technologies and GIS techniques, many location-aware or GPS-based devices have been established.

General issues and techniques of information extraction as well as extracting geographic information such as addresses and location names have been identified in [11, 12, 13 & 14]. Olga [15] describes an algorithm for finding the class of geographic names i.e. city, region, country, island, river, and mountain by using patterns and adding them to a name and sending this query to a search engine. Based on the number of results, the algorithm determines which geographical category is suitable for the name. For example, the number of retrieved documents for a query like *city of Tokyo* is probably much higher than something like *Tokyo country*. Therefore, the algorithm automatically considers Tokyo as a city and puts it in city

¹ <http://local.google.com/>

² <http://local.yahoo.com/>

³ <http://www.northernlight.com/>

⁴ <http://www.geosearch.ca/Index.html>

table. The system is very accurate for all geographical features excluding city names. The method is time-consuming and is not useful for real-time search.

Geographical web services and systems need defining and utilizing ontologies. Such geographical ontologies have been described in [9, 16 & 17]. Watters & Amoudi [3] report on the GeoSearcher project for re-ranking search results based on a location-based approach. GeoSearcher sends a query and a reference point to a search engine and tries to identify the geographic coordinates (latitudes and longitudes) of the webpages by assistance of online gazetteers. Having the coordinates, the algorithm calculates the distance of selected webpages from the reference point. The results are then ranked in ascending order of distance. This approach also is slow and needs various online resources.

Gravano et al. [18] divided Web queries into global and local based on search results. If the best results of a query are local webpages i.e. the information is suitable for a special area, the query is considered a local query; otherwise it is global. For example, the query *wild flowers* seems to be global while the query *real estate* retrieves more local results. The authors used this algorithm to re-rank search engine results for local queries. This method relies on the frequency of location names in the retrieved pages. The system does not focus on a special location as reference point and all location names in a page are considered by the algorithm.

Work on geographic queries has also been reported by Sanderson & Kohler [19]. The authors ran a survey on the Excite Search Engine⁵ to analyse geographic queries. 18.6% of queries had geographic names (including name of mountains, seas, etc.) and 14.8% had location names (cities, states and countries). Geographic queries were classified in different categories and the results show that locations, commerce and services as well as recreation are most common subjects of geographic queries. The study also indicates that geographic queries are often longer than other queries.

3 Methodology

To support this study, we conducted a survey on the MetaCrawler Search Engine⁶ to analyse real queries which people use to search the Web. MetaSpy⁷ is a service of MetaCrawler that allows tracking of current searches. A random set of 4350 queries was extracted from the Exposed MetaSpy which shows all searches including sex and nudity queries. All queries were manually controlled by the researchers and location names were compared with to various resources in the case of ambiguity. Symbols, characters and Boolean features were ignored. A typical analysed entry looks like this:

Query	:	<i>Rental car Illinois</i>
Type	:	<i>lbq</i>
Pattern	:	<i>l</i>
Subject Category	:	<i>car</i>
Location Type	:	<i>state-us</i>
Location Tree	:	<i>N-America/USA/Illinois</i>

⁵ <http://www.excite.com/>

⁶ <http://www.metacrawler.com/>

⁷ <http://www.metaspys.com/info.metac.spy/metaspys/>

Queries written in non-Roman scripts such as Arabic or Chinese were excluded. For similar locations, wherever possible, we considered the bigger or more famous one; otherwise, the same query was sent to MetaCrawler and the location was judged based on the ten first results. Several patterns for location-based queries were defined based on the surveyed queries.

4 Geospatial Queries (GSQ) on the Web

Many webpages have geospatial values and it is possible to assign a location to them. This location might refer to the source [3] or to the target [20]. We define geospatial queries (GSQ) as those containing location names or their subjects refer to special regions in the world. People may add a city, state or country name to their queries. Our survey as shown in Fig. 1 unveils that most of the GSQs are searched in city level. For the USA, the name or abbreviations of states are common. Geospatial queries can be divided into two major categories: local queries and location-based queries.

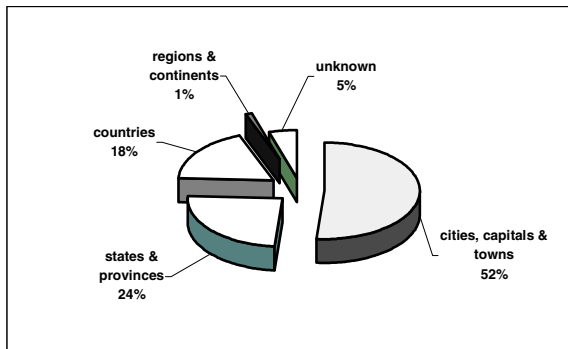


Fig. 1. Different types of location names in search queries

4.1 Local Queries (LQ)

A local query (LQ) asks about natural or human-made features on the Earth or investigates different topics in a specific area. They all contain a geographical or location name in them. The quality of a webpage is defined differently by each search engine; however, the search engine’s ranking algorithm is applied equally to all queries. Similar to normal queries, it is expected that the best results contain more keywords, obtain better proximity of keywords or be linked by high quality webpages. Local queries can be categorized in two groups:

Factual Local Queries. These queries ask about facts, figures, geographical and other information which describe a special location. They might be geographic factual questions which ask about geographical facts in the world by use of natural language

[21]. Here are some examples for factual queries: *Italy, What is the biggest country in Africa and the population of New Zealand.*

Subject Local Queries. Unlike the factual queries that ask for descriptions and facts about a location, subject local queries focus much more on a specific topic but in a special area. For example, the topic of a query like *drug addiction in France* is obviously drug addiction but the user is interested to study this subject in relation to France. For subject local queries, the quality of a document is more important than the physical location of it. In other words, the information itself is important not where this information comes from. It is possible that the best results for the above mentioned query is obtained from America or Asia but not France. In this case the user can trust the ranking model of a search engine. Most of the local queries are subject-based, such as: *Soy bean production in china, London crime rate, and Texas oil history.*

4.2 Location-Based Queries (LBQ)

We define a location-based query as one that looks for a service, product, business or professional group in a particular area. Unlike local queries, the site of the provided information is important for location-based queries. For example, if you are looking for *air pollution in Queensland*, what does matter is finding a good report on air pollution in Queensland irrespective of the physical location of the document. A query such as *wedding service, Queensland* seems to be more related to the location. In the latter case, both the quality of information and the location of webpage or the geographical scope of a website is important. The user is probably interested in finding a wedding service in the Australian state of Queensland.

The location of Web resources can be defined in various ways. The location of a webpage can be considered as its physical location. For example, GeoSearcher [3] tries to find the IP of servers which host web pages and ranks the search results based on the distance of servers from the user's computer. This is a robust algorithm; however, most of the times, the host server is not necessarily in the same area that Webpage refers to. A user may lose a good website about *dance club in Moscow* if that site is hosted by a server in the United States. Therefore, the location of a webpage is better to be extracted through the addresses, metatags, headings and other information in that page or even through the link structure of Web documents. Living needs and daily activities are reflected in the Web queries. People search the Web to find houses, restaurants, car services, jobs and recreation opportunities in their surroundings. So, we can predict that in such cases the best Webpages are the ones that are not only ranked higher by a search engine but are also geographically close to user or to the location which is determined by the user. The following examples can be considered as location-based queries: *Property for sale, Heating and cooling service in Clarksville, Real estate and Cabin rental Illinois.*

4.3 LQ and LBQ Similarities and Differences

At the first glance, it seems to be impossible to determine if a query is location-based or not. The real intention can only be determined by the user. However, there are some criteria that can help us to judge whether a query is LQ or LBQ:

- The subject of a LBQ is limited to daily life requirements such as shopping and accommodation which can be addressed through physical places while LQ includes a variety of human intellectual, scientific, entertainment etc. subjects.
- LQ has necessarily a location name with or without a subject, while LBQ consist of a service with or without a location name.
- LQ could have an adjective as its location, but LBQ usually does not use adjectives. For example, “American movie stars” is more likely to be a LQ rather than a LBQ.

5 Patterns of Location-Based Queries

Nine different patterns for location-based queries have been determined through the manual control of the query set. The patterns are chosen based on the researchers’ judgement. The list of abbreviations and symbols used in defining patterns are available at the Table 1. For example, $Q_l ::= \{S \rightarrow K_s \rightarrow L\}$ means that the location-based query Q_l consists of three elements namely S , K_s and L and \rightarrow indicates the direction or sequence of elements in this pattern. A location-based query often consists of two important components: 1) A service, product, profession or business name (S); and 2) location name (L). As we will show, S is the basic part of a location-based query.

Table 1. Abbreviations and symbols used to define the patterns

Symbol	Meaning
Q_l	“location-based query”
S	A service, product, profession or business name, such as child care, restaurant, dentist, pizza, child care and so on
L	A location name including suburb, city, state and country etc. names
K_s	A set of one or more keywords which represents a spatial relation between S and L. Examples: in, at, close, far, around, between, etc.
K_d	A set of one or more keywords which shows compass or geographical directions. Examples: west, southern, north east
A	Any type of address, e.g. street name, zipcode, building number
P	Possessive prepositions: (of) or (’)
::=	“ is defined as”
\rightarrow	direction or order between two elements. $L \rightarrow S$ means that S is located after L
{ }	Indicates that Q_l consists of a set of elements
< >	Indicates that the element is optional

The following patterns are high-frequent in location-based queries. Fig. 2 shows the order of elements in location-based queries in LBQ patterns.

5.1 Omission of Extra Keywords

This is a common pattern for LBQs. People are used to enter a service name S and a location name L without extra keywords or more information (patterns 1 and 3). However, S and L can be divided into two sections (patterns 2 and 4):

- Pattern 1: $Q_l ::= \{S \rightarrow L\}$
 Pattern 2: $Q_l ::= \{S \rightarrow L \rightarrow S\}$
 Pattern 3: $Q_l ::= \{L \rightarrow S\}$
 Pattern 4: $Q_l ::= \{L \rightarrow S \rightarrow L\}$

Examples: *New York cabs, Locksmith Melbourne, Cheap takeaway TX, Real estates, Michigan rental vacation, Oswego used car NY*

5.2 Spatial Relationship

Many queries have one or more words in them which indicate relations between S and L. Often K_s is *in*; however, other prepositions and words such as *at, close, near, around* etc. are used frequently. The query is constructed as:

$$\text{Pattern 5: } Q_l ::= \{S \rightarrow K_s \rightarrow L\}$$

Examples: *Car rent in NY, cheap accommodation in Hawaii, Motels around Tokyo*

5.3 Geographical Directions

Location names accompanied with compass directions (K_d) can make location-based query patterns. North, Southern and South West are examples of directions. The pattern looks like:

$$\text{Pattern 6: } Q_l ::= \{S \rightarrow \langle K_s \rangle \rightarrow K_d \rightarrow \langle OF \rangle \rightarrow L\}$$

Examples: *Back packers north of Queensland, car delivery in the western London, Thai food around south east of Dakota*

5.4 Possession

Many queries follow the rules of natural languages. Possessive prepositions (P) especially OF are used in many location-based queries:

$$\text{Pattern 7: } Q_l ::= \{S \rightarrow \text{"OF"} \rightarrow L\}$$

Examples: *Hotels of Esfahan, Ethnic butcheries of Calgary*

5.5 Other Patterns

Service Names. Users might use a subject without entering a location name as a reference point. If the subject is a service or business, it is probably a location-based query. This pattern is very common:

$$\text{Pattern 8: } Q_l ::= \{S\}$$

Examples: *List of summer pools, Real estate, Buy coral calcium, Garage sale*

Addresses and Places. Adding an address (A) such as street address, zip code, building name etc. to a subject indicates that probably the user is interested in a location-based search. In combination with a service or facility name they can make an LBQ:

$$\text{Pattern 9: } Q_l ::= \{S \rightarrow \langle K_s \rangle \rightarrow \langle K_d \rangle \rightarrow A \rightarrow L\}$$

Examples: *Oxford Street London shopping, Car service post code 3233*

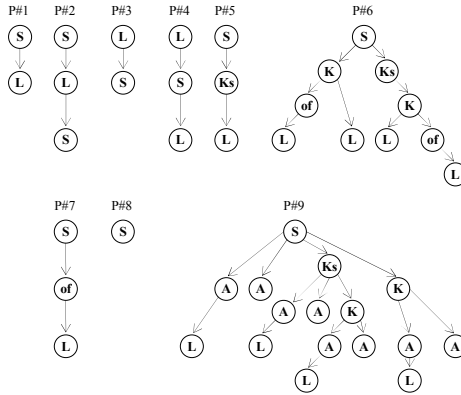


Fig. 2. Order of the components in location –based queries

6 Discussion

To find the portion of geospatial queries among all Web queries, we added the number of queries containing location names to the number of location-based queries which followed pattern number 8. Table 2 indicates that 14.16% of queries contain location names. Also, more than one third of geospatial queries do not have any location name in them. Altogether, 22.67% of all queries can be considered as geospatial queries which supports the necessity of establishing and improving search facilities for this sort of queries.

Table 2. The portion of geospatial queries (GSQ) in the studied dataset

Data set	Number	Percent
GSQ	986	22.67%
GSQ containing location names	616	14.16%
GSQ without location names	370	8.51%

More than half of the location names are capitals, big cities or small towns (Fig. 1). States and provinces are ranked second and they often include the full name or abbreviation of U.S states.

6.1 Evaluation

The dataset was used to measure the coverage of the above-mentioned patterns. Table 3 shows the results of the evaluation. Among the 4350 queries of the dataset, 686 queries (15.76%) have been considered as location-based queries. The last column of the Table 3 surprisingly indicates that about 54% of location-based queries follow pattern 8. In other words, people usually use location-based queries without adding a location. This fact emphasises the importance of developing various methods and tools for determining location-based queries automatically.

Table 3. Distribution of the Location-Based Queries Patterns in the dataset

Pattern	Matched in total set	Matched in LBQ set
1	1.9%	12.40%
2	0.14%	0.87%
3	3.91%	24.78%
4	0.16%	1.02%
5	0.91%	5.83%
6	0.02%	0.15%
7	0.14%	0.87%
8	8.51%	53.93%
9	0.02%	0.15%
Total	15.76%	100%

Patterns 1, 3 and 9 cover more than 90% of all location-based queries. They contain only a service name with or without a location name. As a result, it is possible to distinguish and control most of the location-based queries through having access to yellow pages and databases of location names. A comparison with Sanderson's and Kohler's work (Table 4) shows that the proportion of queries with location names in both studies is similar.

Table 4. Comparison between the results of Sanderson's & Kohler's study and the study here on queries with location names

Item	Sanderson study	Current study
Portion in the total set	14.8%	14.16%
Average length of queries	3.4	3.0
1 term	9.5%	2.7%
2 terms	27.6%	22.7%
3 terms	24.7%	34.6%
4 terms	16.8%	25.8%
5 terms and more	21.4%	14.2%

6.2 Categories of Subjects

Table 5 illustrates different categories of services and products in LBQs. Most of the LBQs look for dealers or services of cars, mobiles, electronic devices etc.

Table 5. Categories of location-based services in Web search queries

LBQ Subject Category	Total Set	LBQ Set
Products	2.31%	14.72%
Public Services	2.09%	13.26%
Travel & Tourism	2.03%	12.97%
Accommodation	1.79%	11.38%
Governmental Services	1.26%	8.02%
Car	1.22%	7.73%
Education	1.19%	7.58%
Health	1.15%	7.29%
Food	0.85%	5.39%
Employment	0.69%	4.08%
Lawyers	0.59%	3.79%
IT	0.43%	2.77%
Other	0.16%	1.02%
Total	15.76%	100%

6.3 Ambiguities

The patterns described above can help to determine location-based queries. However, it is not possible to differentiate between LBQs and other queries completely without the user's direct assistance. There are several important ambiguities:

Service or Subject. It is hard to understand whether a subject is referring to a service and business or if it is asking about a general topic. For example, the query "real estate" can be a LBQ if the user is searching to find accommodation; or it might be a simple request for facts and information about the profession of real estate.

Location Name Ambiguity. Many locations in the World have similar names. For example, Newcastle is a city in England as well as in Australia. Sometimes, the name of a location is similar to a person or any other proper name, for example Washington. The name of a city can be identical or similar to the name of a country, province or state such as Singapore, Tehran and New York. Location names in different languages follow different rules and the main challenge is proper transcription. This is a big issue for the Internet; because the content of the Web is developed by numerous people without any linguistic control.

User Intention. A query can be similar to an LBQ even though the user is asking for general information or a user may ask a general query while he is interested in location-based information. A location name in a query is not enough to consider it as an LBQ. The easiest solution is asking the user whether he is interested in location-based information or not. In the Google Local interface for example, the location input is simply separated from the service name. However, these tools have a limited regional target and most people still prefer to use general search engines rather than location-based facilities.

7 Conclusion

In this paper, we reviewed the specifics of geospatial queries on the Web that include more than 22% of search queries. While 14% of Web queries have at least a location name in them, they cannot easily be judged as location-based. We discussed earlier that location-based queries are intended to search for products and services related to specific locations. In spite of increasing demand for location-based search, tools are insufficient in both availability and quality. Although some spatially-aware search tools have been developed, people use general search engines for location-based purposes. In the future, we plan to examine algorithms to determine location-based queries automatically based on the described patterns. This can help search engines to distinguish such queries and treat them properly based on their geospatial dimensions.

References

1. Intelligence - Center. Google en chiffres: Les données clés sur la société, les hommes, les machines. (2004). English translation retrieved April 5, 2005, from: <http://www.google.angel-cage.de/html/newsstatistics0704.html>.
2. Spink, A. et al. From e-sex to e-commerce: Web search changes. *IEEE Computer*, 35 (3) (2002): 133-135.

3. Watters, C. & Amoudi, G. GeoSearcher: location-based ranking of search engine results. *JASIST*, 54(2) (2003): 140-151.
4. Martin County GIS Team. GIS GLOSSARY. (2003). Available at: www.martin.fl.us/GOVT/depts/isd/gis/glossary.html [Last visit July 22, 2005].
5. MaCGDI. What's spatial data? (2004). Available at: http://www.nalis.gov.my/website/web/html/faqs_info.htm [Last visit July 22, 2005].
6. NCGE. The five themes of geography. National Geographic Education. (1984). Available at: <http://www.nationalgeographic.com/resources/ngo/education/themes.html> [Last visit July 22, 2005].
7. Yet Another Internet Dictionary. (2004). Available at: <http://dpsinfo.com/help/words.html> [Last visit July 22, 2005].
8. Newcomb, K. Google Gets Local in Canada. *ClickZ News*, Sep. 23 (2004). Available at: <http://www.clickz.com/news/article.php/3411681> [Last visit July 22, 2005].
9. Jones, C. B. Spatial Information Retrieval and Geographical Ontologies: An Overview of the SPIRIT Project. *Proc. of CIGIR, Tampere, Finland* (2002): 387-8.
10. Stanley, J. W. & Speights, W.S. Website localization. *Proc. of the Annual Int'l Conference on Computer Documentation, New Orleans, USA* (1999): 127-130.
11. Soderland, S. Learning to extract text-based information from the World Wide Web. In the *Proceedings of SIGKDD* (1997).
12. Muslea, I. Extraction patterns for information extraction tasks: A survey. *AAAI Workshop on Machine Learning for IE, Orlando, USA* (1999).
13. Li, H., et al. Location normalization for information extraction. *Proceedings of 19th Int'l Conference on Computational Linguistics, Taipei* (2002): 549-555.
14. Buyukkokten, O., et al. Exploiting geographical location information of Web pages. *Proc. of the SIGMOD WebDB'99, Philadelphia, USA* (1999).
15. Olga, O. Extracting Geographical Knowledge from the Internet. *Proceedings of ICDM-AM International Workshop on Active Mining, Maebashi, Japan* (2002).
16. Fonseca, F. T. & Egenhofer, M. J. Ontology-driven geographic information systems. In the *Proceedings of ACM GIS'99, Kansas City, USA* (1999): 14 – 19.
17. Jones C.B., et al. Maintaining ontologies for geographical information retrieval on the web. In Meersman, R.; Tari, Z.; Schmidt, D. C. (Eds.) *On the Move to Meaningful Internet Systems, ODBASE'03, Catania, Italy* (2003): 934-951.
18. Gravano, L., et al. Categorizing Web Queries According to Geographical Locality. *CIKM, New Orleans, USA* (2003).
19. Sanderson, M. & Kohler, J. Analyzing geographic queries. *Proceedings of Workshop on Geographic Information Retrieval SIGIR, Sheffield, UK* (2004).
20. Ding, J., Gravano, L. & Shivakumar, N. Computing geographical scopes of Web resources. *Proceedings of the 26th VLDB Conference, Cairo, Egypt* (2000).
21. Radev, D. R., et al. Mining the web for answers to natural language questions. *Proceedings of the CIKM, Atlanta, Georgia, USA* (2001): 143-150.

Self-adaptive Statistics Management for Efficient Query Processing

Xiaojing Li, Gang Chen, Jinxiang Dong, and Yuan Wang

College of Computer Science, Zhejiang University, Hangzhou, 310027, China
{xjli, cg, djx}@zju.edu.cn

Abstract. Consistently good performance required by mission-critical information systems has made it a pressing demand for self-tuning technologies in DBMSs. Automated Statistics management is an important step towards a self-tuning DBMS and plays a key role in improving the quality of execution plans generated by the optimizer, and hence leads to shorter query processing times. In this paper, we present SASM, a framework for Self-Adaptive Statistics Management where, using query feedback information, an appropriate set of histograms is recommended and refined, and through histogram refining and reconstruction, fixed amount of memory is dynamically distributed to histograms which are most useful to the current workload. Extensive experiments showed the effectiveness of our techniques.

1 Introduction

Mission-critical information systems require consistently good performance. To this end, virtually all large databases are managed by well paid system administrators. The administrators must consistently monitor workload and system performance, predict unforeseen or detect existing problems and give corresponding reaction. However, the task has become more challenging due to increasing complexity and skilled system administrators have become more scarce and expensive to employ. This situation calls for a new generation of self-tuning database technologies. Over the past decade, the importance and urgency of self-tuning database technologies have been deeply understood. Gerhard Weikum in his brilliant work [1] showed the challenges, achievements and direction toward a self-tuning database. Most DBMS vendors provided tools to help automate the process of database administration which now mainly focus on index and materialized view recommendation [2-6], table partition [7], statistics organization and maintenance [8-12].

Statistics, especially histograms, are widely used by the query optimizer of a relational database in choosing good execution plans [13]. The automation of statistics management is of great value in improving system performance. Research work toward automating statistics management can be classified to Static-Workload-Analysis (SWA) approach and Execution-Feedback-Analysis (EFA) approach. SWA approach is solely based on the form of queries in the workload and gives its recommendations for statistics maintenance through analyzing the query workload. The representative example of this approach is given by the SQL Server technique described in [8] and [14]. EFA approach automates statistics recommendation and

refresh by monitoring feedback from the execution engine. Our approach falls into this category. DB2 UDB, Microsoft SQL Server and Oracle 10g all exploit this approach [9]. In DB2 UDB, query feedback information is stored in a QFW. Through analyzing QFW, stale statistics are recognized, and appropriate number of frequent values, column-group statistics are recommended [9]. Moreover, the learning optimizer LEO in DB2 uses the actual selectivity feedback from each operator in the query plan to maintain adjustments that were used to correct the estimation errors from histograms, and doesn't change the histograms themselves [15]. Different from DB2, we recommend histograms not frequent values, and the refining is directly carried on histograms themselves. Self-Tuning histograms proposed by the SQL Server technique described in [16, 17] can use query feedback information to adjust frequencies of buckets in existing histograms. However, later refining will ruin results of former ones as pointed out in [18]. Further more, histogram reconstruction doesn't consider workload information and can only redistribute memory within one histogram, while our technique considers access frequencies in the workload for all histograms and can globally distribute memory to the most critical histograms. Jagadish et al in [19] present several greedy and heuristic algorithms for distributing memory among a set of single attribute histograms, but they doesn't address the problem of finding the sets of attributes to build histogram on as in our technique. Lim et al in [20] also propose a query-feedback-driven variant of the synopsis approach where Junction trees of Ghordal Graphes and Delta Rule are used for histogram recommendation and refining. Though interesting, as pointed out in [9], this approach suffers from the scalability problem. Our former work [18] proposed approaches for construction, refining and selectivity estimation of Self-Learning Histograms (SLH) which can overcome the shortcomings of [16]. In this paper we extend it by addressing the issue of recommending SLH histograms and dynamically adjusting memory distribution among them.

In summary, our contributions are:

- We develop a new method to recommend and maintain an optimal set of SLH histograms using on query feedback information, which avoids scanning the base data, and can well adapt to workload and data distribution changes.
- We propose a new approach which can dynamically and globally distribute memory distribution to the most useful histograms in the workload.
- We carried out extensive experiments to validate the effectiveness of our techniques.

The rest of this paper is organized as follows. In section 2 we give some definitions used in this paper. Section 3 describes query feedback processing and selectivity estimating in detail, which is the main body of this paper. Section 4 shows our experimental results and section 5 summarizes the whole paper.

2 Preliminaries

Consider a relation R , let $R = \{A_1, A_2, \dots, A_k\}$. The domain D_i of each attribute A_i is the set of real values of A_i that are present in relation R . Let D_i be indexed by $\{1, 2, \dots, |D_i|\}$, for each value $d \in D_i$, the frequency $f(d)$ is the number of tuples in

R satisfying $A_i = d$. Let r be a set of real intervals containing attribute A_i , we use $A_i \in r$ to represent a range query on attribute A_i . Generally each range r can be represented by $[low, high)$, where low and $high$ are the lower and higher bound of r respectively. The frequency of range r on relation R is defined to the number of tuples in relation R satisfying $low \leq A_i < high$.

Let $|R|$ denote the total number of tuples in relation R , the selectivity $S(r)$ of a range query $A_i \in r$ is denoted by

$$S(r) = \frac{f(r)}{|R|}$$

which is used by the query optimizer to choose efficient execution plans in a cost-based manner.

A histogram H_i on attribute A_i is an approximation to the frequency distribution of attribute A_i . It is constructed by partitioning the data distribution of A_i into several buckets and approximating the frequencies and values in each bucket. A *SLH histogram* H_β with β buckets $\{B_1, B_2, \dots, B_\beta\}$ is a quadruple (V^*, F^*, C^*, T^*) , where V^* is a vector $\langle v_1, v_2, \dots, v_{\beta+1} \rangle$ which represents boundaries between buckets, F^* is a vector $\langle f_1, f_2, \dots, f_{\beta+1} \rangle$ where each f_i represents a cumulative frequency, namely the number of tuples in R satisfying the predicate $A_i < v_i$, and C^* is a vector $\langle c_1, c_2, \dots, c_{\beta+1} \rangle$ where c_i represents CQF(Code of Query Feedback information), and $c_i = c_j$ means the frequency of range $[v_i, v_j)$ is accurate and it is $f_j - f_i$. Bucket B_i can be represented by $[v_i, v_{i+1})$, and f_i and c_i is the cumulative frequency and CQF of boundary v_i respectively. We noted that this could be represented by a rule $F([v_i, v_j)) = f_j - f_i$, so H_β can also be regarded as a set of rules $\{F([v_i, v_j)) = f_j - f_i \mid c_i = c_j, i \in [1, \beta+1], j \in [1, \beta+1], j \geq i\}$. Rules can be used to infer more statistics. For example, from $F([v_i, v_j)) = 0.6$ and $F([v_i, v_{j-1})) = 0.5$, we can infer $F([v_{j-1}, v_j)) = 0.1$, which means the accurate selectivity of query $v_{j-1} \leq A < v_j$ is 0.1. In [18] we showed how rules can be used to avoid ruining the results of former refining. T^* is a $\beta+1$ dimension vector $\langle \tau_1, \tau_2, \dots, \tau_{\beta+1} \rangle$ where each τ_i reflects the current value of *SCF* (defined later) when the rule is introduced into the histogram. It is used to drop outdated rules from the histogram, as we discussed in [18].

Unit-frequency reflects the average frequency of each value in a bucket B_i based on uniform data distribution assumption. It is denoted by $\frac{f_{i+1} - f_i}{v_{i+1} - v_i}$.

System Change Factor (SCF) represents the changes to data. The initial value of *SCF* is 1, later when insertion, deletion and update come, *SCF* will be added by the percentage of modified tuples. E.g. if 20% tuples are deleted, *SCF* will increase 0.2.

A *query feedback* is a quadruple $FB(A_i, low, high, n)$ where A_i is the attribute accessed, low and $high$ represent the lower and higher bound of the query range respectively, and n is the actual number of tuples satisfying the query $low \leq A_i < high$.

3 SASM

3.1 Framework Overview

In this section, we propose SASM, a general framework aimed at recommending and maintaining a set of SLH histograms within fixed amount of memory. SASM monitors query feedback, and by analyzing estimation errors of queries, gives its recommendations. When statistics memory limit is reached, SASM dynamically reclaim some buckets from non-essential histograms and allocate them to the most critical histograms. The workflow of SASM is shown in Fig. 1.

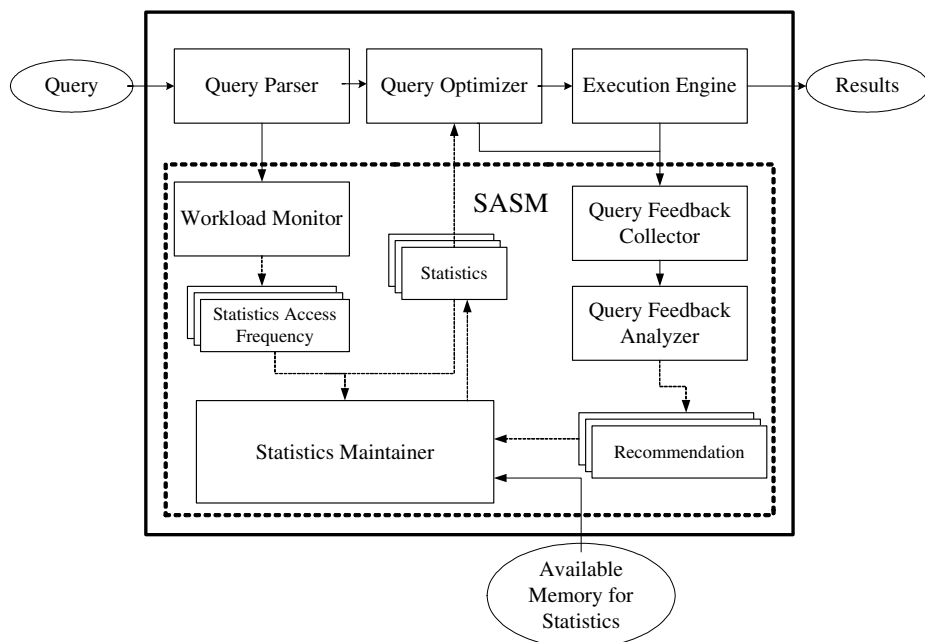


Fig. 1. Framework of SASM

Workload Monitor (WM) monitors how frequently each attribute is accessed. It maintains an Access-Counter for each attribute. The Access-Counter will be used to dynamically distribute memory among all histograms which we will describe in subsections 3.3. The output of WM is recorded in Statistics Access Frequency (SAF) which is organized in a table. For each column, the table it belongs to, and the corresponding Access-Counter are stored in the table as a tuple. Attributes absent from the table means that they are not frequently accessed in the current query workload, and it is non-essential to maintain statistics for them.

When the Query Optimizer (QO) chooses the best execution plan for a query based on costs of different ones, the size of the query, namely the number of tuples satisfying the query predicates, is estimated by its cost-estimation model. And when

the chosen plan is executed by the Execution Engin (EE), the actual query size can be gained. Query Feedbck Collector (QFC) collects these kinds of information together with the skeleton of the execution plan. Table 1 shows an example of the information QFC collects. The first tuple in Table 1 means the estimated and actual output size of query $18 \leq Age < 20$ is 9485 and 8945 respectively, and the estimated output size (9485) is obtained using a histogram.

Table 1. Query Feedback Information

Table	Attribute	Low	High	useHist	Est	Actual
Student	Age	18	20	Y	9485	8945

The query feedback information collected by QFC is then analyzed by the Query Feedback Analyzer (QFA). QFA computes the estimation error for the query and judges whether the estimation error exceeds a given threshold φ . If exceeds, the attribute referenced by the query is recommended as a candidate column for statistics building or refining, else it is ignored. The estimation error used is relative error defined as follows.

$$\delta = \frac{|\sigma - \sigma'|}{\max(\sigma, 1)} \quad (1)$$

where σ' and σ denotes the estimated and actual query size respectively. The recommendation information is similar to that showed in Table 1.

Recommendation for statistics building or refining given by QFA may all ask for memory to build new histograms or accommodate new buckets. If available statistics memory is not enough, it must be reclaimed from non-essential histograms. Statistics Maintainer (SM) is responsible for these issues. It uses information in SAF table and existing histograms to determine from which histograms to reclaim memory and build a new histogram or refine an existing histogram using query feedback information that QFC collects. In section 3.2 and 3.3, we will show how SM works in detail.

3.2 Candidate Statistics Recommendation and On-line Refining

In this section, we show how to use query feedback information to recommend candidate histograms and to refine existing histograms. First we consider there is enough memory. Suppose the query feedback information on A_i in R is $FB(A_i, X, Y, n)$. When memory is enough and no histogram exists on A_i , a new histogram is built using $FB(A_i, X, Y, n)$. We initialize the new histogram to $H_i = (V^*, F^*, C^*, T^*) = (< X, Y >, < 0, n >, < 0, 0 >, < SCF, SCF >)$. H_i and other existing histograms form the set of candidate histograms. Later H_i will be refined when accessed and the estimation error exceeds the given threshold φ . A histogram H_i may already exist on A_i , then it is just refined using algorithm *HistogramRefining* in Fig.2.

Input: $FB(A_i, X, Y, n)$
Output: newly built H_i or H_i refined by $FB(A_i, X, Y, n)$
BEGIN
Find buckets B_i and B_j where $v_i \leq X < v_{i+1}$, $v_j \leq Y < v_{j+1}$
If X (or Y) doesn't equal to any boundaries of B_i (or B_j)
 split B_i (or B_j) to $[v_i, v_a)$ and $[v_a, v_{i+1})$ (or $[v_j, v_b)$ and $[v_b, v_{j+1})$) where
 $v_a = X$ (or $v_b = Y$)
Compute $f_a = \frac{X - v_i}{v_{i+1} - v_i} * (f_{i+1} - f_i) + f_i$ (or $f_b = \frac{Y - v_j}{v_{j+1} - v_j} * (f_{j+1} - f_j) + f_j$)
Set c_a (or c_b) to a value different from any existing CQFs
Set $\Delta = f_a + n - f_b$ // estimation error using histogram H_i
if ($c_a = c_1 \parallel c_a = c_{\beta+1}$) {
 for each CQF c_i in H_i , if ($c_i = c_b$), set $f_i = f_i + \Delta$; }
else {
 for each CQF c_i in H_i , if ($c_i = c_a$), set $f_i = f_i - \Delta$; }
Set c' to a value different any existing CQFs expect c_a and c_b ;
For each CQF c_i in H_i , if ($c_i = c_a \parallel c_i = c_b$), set $c_i = c'$;
Set τ_a and τ_b to the current value of SCF , namely set $\tau_a = \tau_b = SCF$;
END.

Fig. 2. HistogramRefining

3.3 Dynamic Memory Distribution Among Histograms

In SASM, global memory redistribution is achieved by splitting and merging buckets among all histograms. Bucket splitting is processed when refining histograms using query feedback information when available statistics memory is enough, which has been discussed in section 3.2. When available statistics memory is not enough, SM will reclaim memory from non-essential histograms by merging some adjacent buckets in them. In this section, we focus on this issue by determining which histograms are non-essential and how to reclaim buckets from them. Merging may cause accuracy losses. Ideal merging tactic should lower the loss to the least.

Merging tactic based only on frequencies as that in [16] is unsatisfying. In [16], adjacent buckets with similar frequencies are merged. However, see Fig.3, the frequency difference between B_i and B_{i+1} ($|10-90|=80$) is much larger than that between B_j and B_{j+1} ($|10-9|=1$), which means the later two buckets are more similar in frequency than the former two. However, merging B_j and B_{j+1} will lead to high accuracy loss, e.g. before merging, the selectivity of query $1 \leq A_2 \leq 10$ is $\frac{10}{|R|}$, but after

merging, the selectivity changes to $\frac{10}{100 \times |R|} \times (10 + 9) = \frac{1.9}{|R|}$. While merging B_i and B_{i+1} can lead to no accuracy loss, for data in them is uniformly distributed and the frequency of each value in the two buckets is always 1 no matter before or after merging.

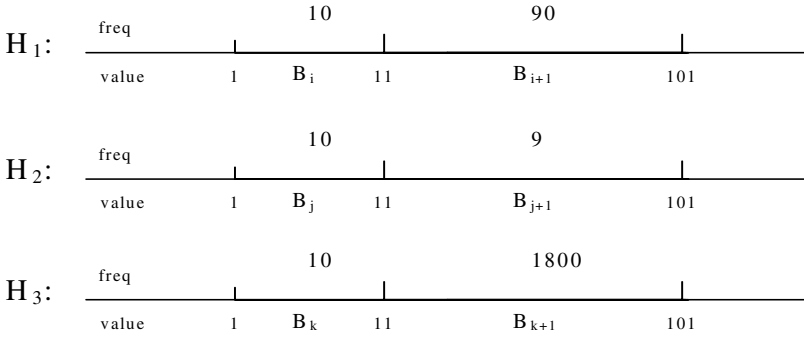


Fig. 3. Histograms

Merging adjacent buckets with similar *unit-frequency* seems a good choice. But since we want to lower the estimation errors for the current running workload, only *unit-frequency* can't meet this goal. E.g. the *unit-frequency* difference of bucket B_j and B_{j+1} is $|10 \div (11 - 1) - 9 \div (101 - 11)| = 0.9$, much smaller than that of B_k and B_{k+1} which is $|10 \div (11 - 1) - 1800 \div (101 - 11)| = 19$. However, if the Access-Counter of H_2 and H_3 is 100 and 1 respectively, then merging bucket B_k and B_{k+1} is better than B_j and B_{j+1} , for H_3 is accessed so seldom, reclaim buckets from it will do little harm of the current workload.

In SASM, we combine the access frequency and *unit-frequency* of histograms when choosing buckets to merge. We compute the Merge-Factor (MF) of two adjacent buckets of all the histograms. If the MF of two adjacent buckets B_i and B_{i+1} is in the $\epsilon\%$ smallest MFs, they are considered to be merged and the histogram they belong to is taken as a non-essential histogram. MF is defined to:

$$MF = Access - counter \times |UF_i - UF_{i+1}| = Access - counter \times \left| \frac{f_{B_i}}{w_i} - \frac{f_{B_{i+1}}}{w_{i+1}} \right| \tag{2}$$

where UF_i , f_{B_i} and w_i stand for the *unit-frequency*, width and frequency(not accumulative frequency) of bucket B_i . It should be emphasized that f_{B_i} and $f_{B_{i+1}}$ are estimated using all related rules as described in [18] with each bucket B_k being seen as a query $v_k \leq A_i < v_{k+1}$.

After merging, B_i and B_{i+1} become to one bucket $B[v_i, v_{i+2})$, the cumulative frequency and CQF of v_i and v_{i+2} are the same as before. If a histogram remains only one bucket due to merging, and its Access-Counter is very low, we can drop it from

the system. In this way, the origin set of candidate histograms is shrunken and memory is dynamically distributed to the most critical histograms.

4 Experimental Evaluation

4.1 Setup for Experiments

We made comparisons in accuracy and maintenance cost between SASM and other approaches including Self-Tuning, MaxDiff, Equi-Depth histograms. We use a database consists 10 tables with each table having more than 5 randomly generated attributes. A Zipfian distribution [21] generator is used to generate skewed data for columns in the database. The degree of skew in the data is controlled by the Zipfian parameter z , varied between 0 (uniform) and 4 (highly skewed). Which attribute a histogram is built on is randomly chosen to simulate the initial status of the workload.

Table 2. Parameter Setting in the Experiments

Experiments No	DSk	WS	MT	SL	DSc
1	1	1	changing	1800Byte	10*100K tuples
2	1	1	0.2	changing	10*100K tuples
3	1	changing	0.2	1800Byte	10*100K tuples
4	changing	1	0.2	1800Byte	10*100K tuples
5	1	1	0.2	1800Byte	changing
6	1	1	0.2	changing	10*100K tuples

Query workload consisted 6000 queries of the form $x \leq A_i < y$, $A_i \geq x$ and $A_i < y$, where x and y were generated. Not all attribute are accessed equally, and the access frequency of each attribute is also controlled by the Zipfian data generator. For example, a class Zipfian(1, nATTs, nQueries, nATTs, z) can distribute nQueries to nATTs attribute with a skew z .

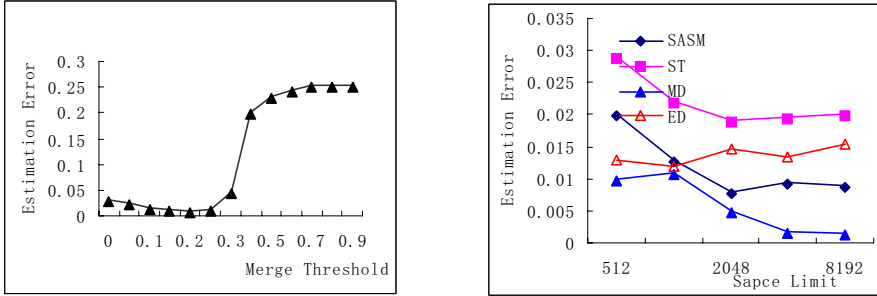
When the query workload was executed, estimation errors and maintenance costs were computed for each approach. We used both average absolute error E_{abs} and average relative error E_{rel} as accuracy metric in evaluating the accuracy of each approach in estimating range query result size. They were computed as:

$$E_{abs} = \frac{1}{N} \sum_{i=1}^N |\sigma_i - \sigma'_i| \text{ and } E_{rel} = \frac{1}{N} \sum_{i=1}^N \frac{|\sigma_i - \sigma'_i|}{\max(1, \sigma_i)} \quad (3)$$

where N represents the number of queries in the workload, σ_i and σ'_i represent the actual and estimated output size of the i th query in the workload. Since results using both error metric show no intrinsic difference, we only presents experimental results using average relative error in this paper in the interest of space.

4.2 Results

We have done extensive experiments to evaluate the effectiveness of our technique under various kinds of situations, but due to limited space, we only show results of six representative experiments. We first give the parameter setting of each experiment in Table 2. DSk, WS, MT, SL and DSc mean Data Skew, Workload Skew, Merge Threshold, Space Limit and Data Scale respectively. Each figure corresponds to the result of one experiment using parameters shown in one tuple in Table 2.



(a) Error - Merge Threshold

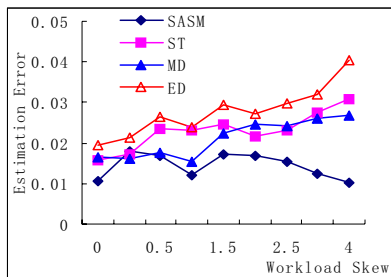
(b) Estimation Error via Space Limit

Fig. 4. Estimation Error via Merge Threshold and Estimation Error via Space Limit

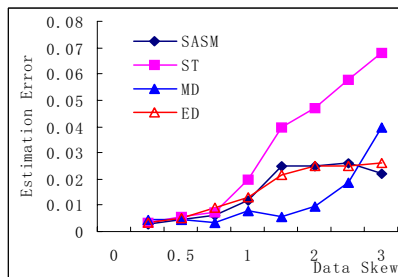
Fig.4.(a) shows the estimation error of SASM changes with the Merge Threshold. ST, MD and ED stands for Self-Tuning histograms in [16], Maxdiff histograms and Equi-Depth histogram respectively. SASM is our approach. From it, when Merge Threshold increases but keeps smaller than 20%, the accuracy of SASM improves, since more memory can be globally distributed to most critical histograms. However when the threshold keeps increasing, the accuracy drops drastically. This is because when Merge Threshold becomes very large, the number of buckets merged in each redistribution of memory among all histograms will be very large and those reclaimed buckets are not all assigned to histograms at once, so space currently occupied by SASM is reduced. From the figure we can see 20% is an ideal value for Merge Threshold, so we will use this value in later experiments. Fig.4. (b) shows the estimation error changes with space limit. Generally, when space limit is large, more buckets can be assigned to each set of histograms and hence they can achieve good approximation to data. Fig.4. (b) shows this trend. When space limit increases, estimation errors of all sets of histograms decrease. When space limit is large enough, all errors keep in a nearly stable state. This is because when existing space is enough to realize the data distribution, more memory can lean to no improvement in accuracy. SASM performs better than ED and ST, while poorer than MD.

Fig.5. (a) shows the estimation error changes with increased workload skew of each set of histograms. We can see that when the workload skew increases, the estimation error all increases except SASM. When workload skew becomes large enough (about $z=2$), estimation error of SASM begins to drop. This is because only SASM can globally distribute memory among all histograms. When workload skew is large, buckets from non-essential histograms are reclaimed and allocated to those

heavily accessed histograms. Fig.5.(b) shows the estimation error changes with data skew. It can be seen that when data skew is small ($z < 1$), each histogram is relative accurate. MD is the most accurate. SASM is nearly as accurate as ED. When data skew increases, the accuracy of all histograms drops slowly. When data skew is very large, the accuracies of MD and ST drop drastically as pointed out in [16] while that of SASM increases instead. This is because when data skew is large, the number of distinct values becomes very small, and by self-refining using query feedback information, SASM can well adapt to this change.

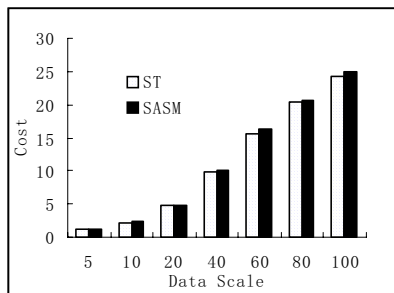


(a) Estimation Error-Workload Skew

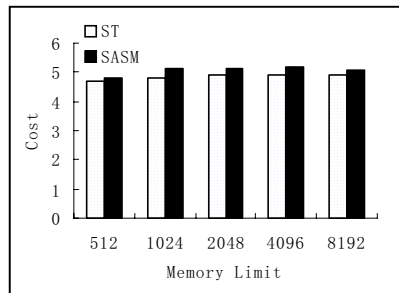


(b) Estimation Error-Data Skew

Fig. 5. Estimation Errors via Workload Skew and Estimation Errors via Data Skew



(a) Cost via Data Scale



(b) Cost via Memory Limit

Fig. 6. Cost via Data Scale and Cost via Memory Limit

In this experiment, we want to compare the costs (seconds) for query processing using ST and SASM. Since MD and ED are static histograms, cost comparison between SASM and them is non-essential. Fig.6. (a) shows the cost of processing the workload changes with the total number of tuples (measured in thousand, e.g. 5 represents 5000 tuples for each table) in the database. We can see that the both costs increase as data becomes larger and the cost of SASM is nearly the same as that of ST. Fig.6. (b) is the result of costs changes with memory limit. The cost of SASM is slightly higher than that of ST. However, since both ST and SASM piggyback on the process of query answering, the cost is very small contrast to the cost of query execution. We can also see from Fig.6.(b) that both costs keep almost stable, this is due to decreases in reconstruction frequency when memory limit is large enough.

5 Conclusion

In this paper, we presented a unified framework SASM for autonomic statistics management in DBMSs. SASM not only can recommend, construct and refine SLH histograms using only query feedback information without accessing the base data, but also can dynamically distribute fixed memory to the most critical histograms. Histogram recommendation and refining are based on estimation error analysis, which makes SASM adapt well to workload and data changes and using CQF, statistics can be deduced using rules and ruining former reefing by later ones is avoided. Experiments showed SASM can reduce estimation errors to a satisfying degree while keep acceptable maintenance cost.

References

1. Gerhard Weikum, Axel Moenkeberg, Christof Hasse and Peter Zabback. Self-tuning Database Technology and Informative Services: from Wishful Thinking to Viable Engineering. Proceedings of the 28th International Conference on Very Large DataBases, HongKong, China, 2002, pp. 20-31.
2. Chaudhuri, S., Datar, M., Narasayya, V. Index selection for databases: a hardness study and a principled heuristic solution. IEEE Transactions on Knowledge and Data Engineering, Vol 16, Issue 11, Nov, 2004, pp.1313-1323.
3. Daniel C. Zilio, Calisto Zuzarte, Guy M. Lohman, Roberta J. Cochrane, Jarek Gryz, Eric Alton and Gary Valentin. Recommending Materialized Views and Indexes with the IBM DB2 Design Advisor. Proceedings of the International Conference on Autonomic Computing, New York, USA, 2004, pp.180- 187.
4. Automated Selection of Materialized Views and Indexes for SQL Databases. Proceedings of the 26th International Conference on Very Large DataBases, Cairo, Egypt, 2000. pp. 496-505.
5. Surajit Chaudhuri and Vivek Narasayya. An Efficient, Cost-Driven Index Selection Tool for Microsoft SQL Server. Proceedings of the 23rd International Conference on Very Large DataBases, Athens, Greece, 1997, pp. 146-155.
6. Kai-Uwe Sattler, Eike Schallehn and Ingolf Geist. Autonomous Query-Driven Index Tuning. Proceedings of the International Database Engineering and Applications Symposium, 2004, pp.439- 448.
7. Sanjay Agrawal, Vivek Narasayya and Beverly Yang. Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design. Proceedings of ACM SIGMOD International Conference on Management of Data, Paris, France, 2004, pp. 359-370.
8. Surajit Chaudhuri, Vivek Narasayya. Automating Statistics Management for Query Optimizers. Proceedings of the 16th International Conference on Data Engineering, 2000.
9. A.Abounaga, P. Haas, M. Kandi, S. Lightstone, G. Lohman, V. Mark, I. Popivanov, and V. Raman. Automated Statistics Collection in DB2 UDB. Proceedings of the 30th International Conference on Very Large Data Bases, Toronto, Canada, 2004, pp.1146-1157.
10. Ihab Ilyas, Volker Markl, Peter J.Haas, Paul G. Brown and Ashraf Abounaga. Automatic Relationship Discovery in Self-Managing Database Systems. Proceedings of the International Conference on Autonomic Computing , 2004.

11. Ihab F. Ilyas, Volker Markl, Peter J. Haas, Paul G. Brown and Ashraf Aboulnaga. CORDS: Automatic Generation of Correlation Statistics in DB2. Proceedings of the 30th International Conference on Very Large Data Bases, Toronto, Canada, 2004, pp.1341-1344.
12. Ihab F. Ilyas, Volker Markl, Peter Haas, Paul Brown and Ashraf Aboulnaga. CORDS: Automatic Discovery of Correlations and Soft Functional Dependencies. Proceedings of ACM SIGMOD International Conference on Management of Data, Paris, France, 2004.
13. Yannis Ioannidis. The History of Histograms. Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 2003, pp.19-30.
14. Nicolas Bruno and Surajit Chaudhuri. Exploiting Statistics on Query Expressions for Optimization. Proceedings of ACM SIGMOD International Conference on Management of Data, Madison, Wisconsin, USA, 2002, pp.263-274.
15. Michael Stillger, Guy Lohman, Volker Mark, Mokhtar Kandil. LEO-DB2's Learning Optimizer. Proceedings of the 27th International Conference on Very Large Data Bases, Roma, Italy, 2001, pp. 19-28.
16. Ashraf Aboulnaga and Surajit Chaudhuri. Self-Tuning Histograms: Building Histograms without Looking at Data. Proceedings of ACM SIGMOD International Conference on Management of Data, 1999, pp. 181-192.
17. Nicolas Bruno, Surajit Chaudhuri, Luis Gravano. STHoles: A Multidimensional Workload-Aware Histogram. Proceedings of ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA, USA, 2001, pp.211-222.
18. Xiaojing LI, Bo Zhou and Jinxiang Dong. Self-Learning Histograms for Changing Workloads. To appear in the Ninth International Database Engineering and Applications Symposium, Montreal, Canada, 2005.
19. H.V. Jagadish, Hui Jin, Beng Chin Ooi and Kian-Lee Tan. Global Optimization of Histogram. Proceedings of ACM SIGMOD International Conference on Management of Data, Santa Barbara, California, USA, 2001, pp.223-234.
20. Lipyeow Lim, Min Wang and Jeffrey Scott Vitter. SASH: A Self-Adaptive Histogram Set for Dynamically Changing Workloads. Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 2003, pp. 369-380.
21. G. Zipf. Human Behavior and the Principle of Least Effort. Addison Wesley, 1949.

Design and Implementation of the Modified R-Tree Structure with Non-blocking Querying*

Myungkeun Kim¹, Sanghun Eo¹, Seokkyu Jang¹, Jaedong Lee², and Haeyoung Bae³

^{1,3}Dept. of Computer Science & Information Engineering, Inha University

²Dept. of Computer Science, Dankook University
{kimmkeun, eosanghun, skjang}@dmlab.inha.ac.kr,
letsdoit@dku.edu, hybae@inha.ac.kr

Abstract. In highly concurrent field such as location based services, massive objects are moving concurrently. Due to continuously changing nature of their location, traditional indexes cannot provide the real-time response since query processing is frequently blocked by node-split or region propagation as the locations of objects change. In this paper, the modified R-tree structure with lock-free querying for multi-dimension data, R^{ver} -tree, is proposed. Basically, R^{ver} -tree uses the new versioning technique. When updating data such as key update(region propagation) and index restructure(node-split), it never physically modify original data, rather creates new version for compensating data intactness. Due to data intactness, search operation can access data without any locking or latching by reading old version. In the performance evaluation, it is proven that search operation of R^{ver} -tree is at least two times faster than a previous work.

1 Introduction

The advance in wireless networks and in positioning systems has led to the emergence of location-based services (LBS). LBS support useful and convenient services based on the user's location such as emergency service, driving direction, and buddy finding. Due to rapidly expanding field of location-based services, a large number of service subscribers is continuously moving and sends queries via wireless communication. Applications for location based services must store the current location of the large number of moving user and process the location based query in real-time manner [13, 14]. To index massive moving objects, R-tree structure [4] may be used. However, traditional R-tree structure suffers from poor performance since the query processing is frequently blocked by node-split or region propagation as the locations of objects change.

During the last decade, a number of index methods, which modified the basic structure of R-Tree, have been proposed to maximize the concurrent efficiency of updating and querying, such as Rlink-tree [6], CGiST[7], and [2],[9],[12]. They have tried to improve the query performance by using the lock minimally or by linking the sibling

* This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

nodes like Rlink-tree. In highly concurrent environments, those index methods are not suitable for indexing moving objects since the query processing is frequently blocked even they have tried to minimize the blocking probability.

In this paper, the modified R-tree structure using “the instant versioning technique” is proposed. It does not physically modify the original data, but rather make a new version for compensating the data intactness. Unlike the traditional versioning technique [11], this technique does not keep multiple versions, but it instantly keeps the original data only until the creation of new version is done. That is, the original data is remained intact not for further operations but for operations that are currently accessing it. According to the instant versioning technique, an entry or a node is versioned. The proposed technique make lock-free search operations by reading old version even during the modification of an entry or a node.

The remainder of this paper is organized as follows. Section 2 describes related works and section 3 describes the proposed concurrency technique. Section 4 presents consistency, section 5 proves that the proposed index is the deadlock-free, and Section 6 presents experiments compared to Rlink-tree. Finally section 7 makes a conclusion.

2 Related Works

In this section we present a problem of concurrent operations in the R-Tree. And we explain how the previous techniques solve this problem. Fig. 1 presents the problem of wrong path.

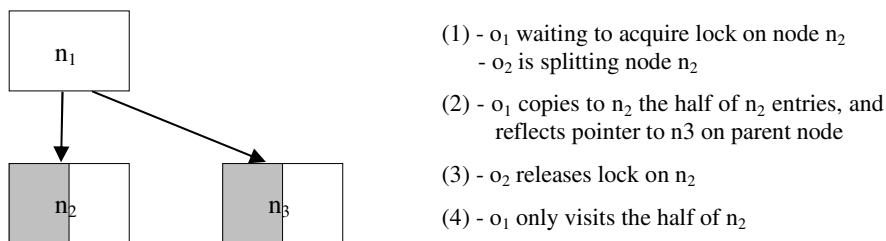


Fig. 1. The problem of wrong path

In Fig. 1, let's assume that o_1 is a search operation that is waiting for acquiring lock on node n_2 , and o_2 is an insert operation that is splitting node n_2 . When o_1 acquires the lock on node n_2 after node-split finishes, it only searches the half of n_2 since the other half of n_2 has been moved to n_3 by o_2 .

The previous concurrency control techniques are classified into the pessimistic solution [2, 9] and the optimistic solution [6, 7, 12] to solve the problem of wrong path. The pessimistic solution does not allow node-splits in their path, and the optimistic solution allows the problem of wrong path but it corrects the wrong path by applying some special action.

The representative technique of the pessimistic solution is the lock-coupling [2, 9]. When descending the tree a lock on a parent node can only be released after the lock on the child node is granted, also when ascending the tree (node-splits or region

propagation) locks on ancestor nodes should be held until ascending step is terminated. This technique decreases concurrent efficiency since minimum of two nodes are kept locked at a time. The optimistic solution needs special method to correct the wrong path and to judge if the visiting node has been modified. The representative technique for the optimistic technique is Rlink-tree [6]. Rlink-tree uses LSN (Logical Sequence Number) to judge if visiting node has been split, and it corrects the wrong path by maintaining the link between sibling nodes. LSN is in charge of same roles as the maximum key of Blink-tree [10] due to the fact that R-Tree has the property that the entries of nodes are not linearly ordered. Each entry in a non-leaf node consists of a key, a pointer, and the expected LSN that it expects the child node to have. If the expected LSN taken from the parent node is different from the actual LSN of child node, a process moves right via sibling link until the node having the expected LSN is found, is carried out.

All of the previous works have the same drawback that search processing should be blocked since they use the lock-based technique. They require the shared mode lock for retrieving the consistent data against update operations. This situation over the lock-based technique cannot be avoided. This paper proposes the version-based technique which enables the search operation to progress without blocking it.

3 R^{ver}-Tree

This section describes the concept of the instant versioning technique, and introduces the modified structure of R-Tree, called R^{ver}-Tree, for applying the instant versioning technique, and describes internal and external operations of R^{ver}-tree. Finally, it discusses the reclamation of garbage space due to the instant versioning technique.

3.1 Instant Versioning Technique

The traditional versioning technique is designed for record manager. This approach can make read-only transaction non-blocking by reading suitable record among multiple versions. From indexing point of view, keeping multiple versions is unnecessary since index operations have to get only the latest versions.

Let $f(x)$ be a function that chooses a latest version among the set of derived versions $\{a^1, a^2, a^3, \dots, a^n, a^{n+1}\}$. a^{n+1} is the derived version from a^n , where n is the version number. If a^n is the latest version and the installation of another new version a^{n+1} is not done yet, the function $f(x)$ chooses a^n . This simple rule can make search operation access data without any locking even when data is being physically modified. Let's assume that a^n is a non-leaf node and it is being split. Node-split does not physically modify a^n , rather makes a new version a^{n+1} . Search operation can traverse sub-tree of a^n via the intact node a^n without any locking. Also this rule is applied to version an entry.

In order to implement the instant versioning technique, the linked list style is used. This technique requires pointers to link between the latest versions of data. Those pointers are not for sorting, but they are for preventing operations from accessing non-latest versions. To add additional link pointers, the traditional R-Tree structure is extended.

First, the entry structure is extended by adding two pointers for linking between sibling entries. Each entry in a node consists of a key rectangle, a pointer to the child node or indexed object, a pointer *nxtActive* to the next sibling active entry, and a pointer *nxtFree* to the next sibling free entry. Also, the node structure is extended by adding two pointers, *fstActive* and *fstFree*, for completing the instant entry versioning. *fstActive* is a pointer that points to the first among the active entries. And *fstFree* is a pointer that points to the first among the unused entry. Search operations initially take *fstActive*, and then they move right via *nxtActive*. They visit only the latest entries since there are only the latest entries in the active link. And update operations take new entry from *fstFree*.

Second, the node structure does not need additional pointers for applying the instant node versioning technique since nodes in the tree are already linked between parent node and child node. Just, the traditional splitting algorithm is modified according to the instant node versioning. A node is instantly versioned only when it is split. In Fig. 3, let's assume that n_2 is splitting. First, new nodes, n_3 and n_4 , are created unlike the traditional splitting (that only creates one new node), then entries of n_2 are distributed onto n_3 and n_4 , then pointer of existing n_2 from parent node is removed, finally the pointers of n_3 and n_4 are inserted to n_1 . This technique does not cause any modification on the splitting node n_2 , so that the search operations can traverse subtree of n_2 without any locking and latching. However, if insert operation is waiting for acquiring lock on n_2 , it may insert a new key to the wrong node n_2 (versioned node). This is the problem of wrong path as mentioned above in the related works. In order to solve this problem, the basic node structure of R^{ver} -tree is extended as described in the following paragraph.

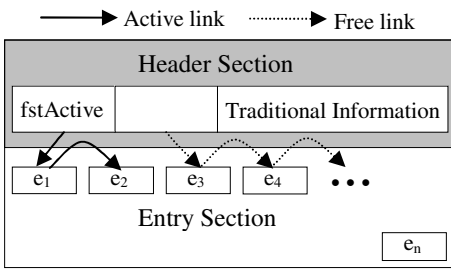


Fig. 2. The basic node structure of R^{ver} -tree

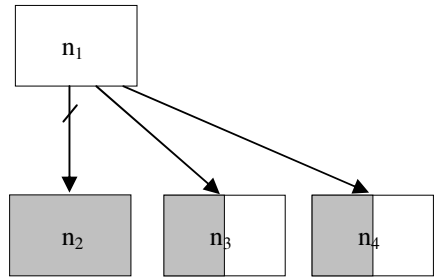


Fig. 3. The instant node versioning

Third, two pointers p_1 , p_2 and a version bit v are added to the basic node structure of R^{ver} -tree. The variables are used to solve the problem of wrong path. That is, operations check v to detect the fact that the node has been split, and use p_1 , p_2 to correct the wrong path. v is marked during the split process, and it indicates the fact that the node is versioned. If a version bit v of visiting node is true, update operations can judge that the node has been split by another operation while they were waiting. p_1 , p_2 are pointers that point to the two new nodes derived from the original node.

If the visiting node is a versioned node, a process moves right via version pointers until meeting the latest nodes. Fig. 5 presents the scenario to correct the wrong path. The version bit v of node n_2 is true, and p_1 , p_2 are pointing node n_3 and n_4 respectively.

The update operation confirms the fact that n_2 has been versioned by checking the version bit v , and fixes the wrong path by moving to n_3 and n_4 through p_1 and p_2 . These pointers may point to another versioned node since the non-versioned node that was derived from versioned node can still be versioned again by subsequent splits, but continued moving via version pointers can guarantee that non-versioned nodes (latest nodes) are found.

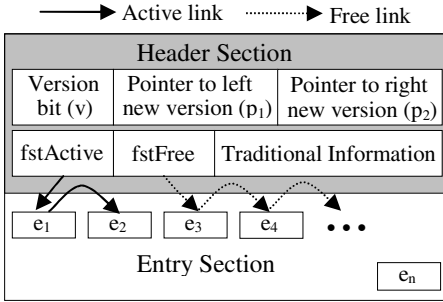


Fig. 4. The extended node structure of R^{ver} -tree

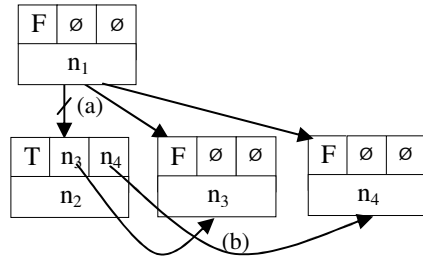


Fig. 5. Correction of wrong path. (a) Detect the fact that n_2 has been split. (b) Visit to new versions via version pointers

3.2 Internal Operations

R^{ver} -tree has four internal operations, such as insertion, deletion, modification of entry, and search operation. They are limited to a node.

Insertion and deletion of entry are very simple. A new entry, allocated from the free link, is appended to the end of the active link. It is linked only after it becomes consistent. If it does not, search operations may access the inconsistent entry. Deletion of entry is done by unlinking from the active link. In Fig. 6 (b), an entry e_2 is deleted by assigning the nextActive of e_1 to e_3 . And then it is linked into the free link. The active link of e_2 should not be cut because of search operations that are accessing e_2 . If it is cut, they cannot move to the next active entry, e_3 . The deleted entries are reused by further operation. It should carefully be reused because of search operations that are still accessing it. In order to simplify the procedure of internal operations, this issue is discussed in subsection 3.4.

Entry modification is a combination of the insertion and deletion of entry. It should be atomically done since search operations could access both entries, the original entry and the entry derived from the original entry. Fig 6 (c) describes modification of the entry e_2 . The entry e_2 is atomically modified by assigning the nextActive of e_1 to e_4 . Especially, when splitting a node according to the instant node versioning, entry modification requires insertion of two entries unlike simple entry modification. It is also done in the same manner.

Internal search operations visit entries by taking nextActive . They do not require any locking and latching due to the atomic linking of internal update operations.

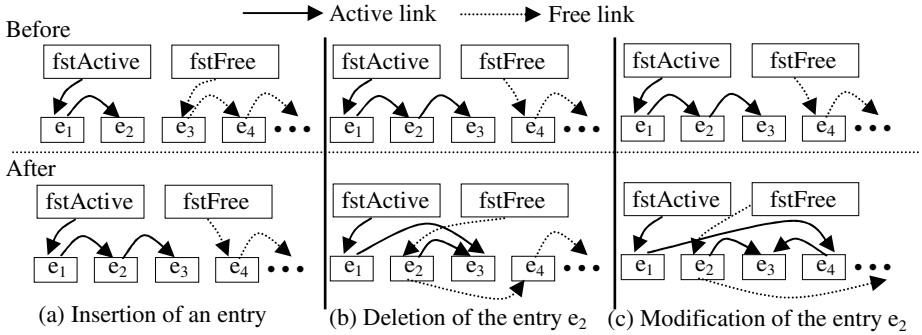


Fig. 6. Internal operations

3.3 External Operations

This subsection describes external operations such as search and insert operation. These operations are interactively invoked by the external component above the index, such as “cursor”. In this subsection, delete operation is not described since it is a combination of the search and insert operation. Deletion of empty node is discussed in subsection 3.4.

3.3.1 The Search Operation

Search operation finds all entries that belong in the range of query condition. Searching starts by pushing the root node pointer to the stack. The stack is used to remember pointers of nodes or objects that need to be tested. The root node pointer will be the first to be popped from the stack. If the popped one is the pointer that indicates the node, all entries qualifying the query conditions in it are pushed to the stack. If it is the pointer that indicates objects, it is returned with the stack to caller who has invoked search operation. In next time, *findNext* procedure is invoked with the returned stack as shown in Fig. 7 (line 6). This process is repeated until the stack becomes empty.

```

1  findFirst(STACK s, RECT r)
2  push(s, root)
3  return findEntry(s, r)
4  end
5
6  findNext(STACK s, RECT r)
7  return findEntry(s, r)
8  end
9
10 findEntry(STACK s, RECT r)
11 while not empty(s)
12   p = pop(s)
13   if(p is pointer to indexed tuple)
14     return p
15   Else
16     for(all entries e of p)
17       if(e is intersecting r)
18         push(s, NODE(e))
19       End
20     end
21   end
22 end
23 end
    
```

Fig. 7. The Search operation

3.3.2 The Insert Operation

The insert operation progresses in three steps. The first step is the descending step. This step finds the optimal leaf node that is fit for inserting the new key by moving down the tree. The second step inserts the new key to the found leaf node, and the last step is the ascending step. It moves up the tree to modify the ancestor nodes in two cases: when the region of node has changed or a node should be split due to the lack of space.

```

1  insert(STACK s, RECT r)
2  findLeaf(s, root, r)
3  n = pop(s)
4  if(n is not full)
5    insert r to n
6    if(MBR(n) has changed)
7      updateParent(s, n, MBR(n))
8    end
9    unlock(n)
10 else
11  s-lock(n)
12  n1, n2 = create new node
13  distribute n's entries to n1 and n2
14  insert a new entry with r to optimal node
15  set p1 and p2 to p's version pointers
16  mark the version bit of p as true
17  splitNode(s, n, n1, n2)
18 end
19 end
20 findLeaf(STACK s, NODE n, RECT r)
21 s-lock(n); push(s, n)
22 if(n is leaf node)
23   w-lock(n)
24 else
25   s-lock(n)
26 end
27 if(n is versioned)
28   pop(n); unlock(n)
29   n = node leading to optimal node
        from new versions of n
30   push(s, n);
31   if(n is leaf node)
32     w-lock(n)
33   else
34     s-lock(n)
35   end
36 end
37 if(n is non-leaf node)
38   e = entry on n leading to
        minimal MBR with r
39   n = NODE(e)
40   findLeaf(s, n, r)
41 end
42 end

```

Fig. 8. The insert operation (descending step)

When descending the tree the visited nodes are pushed to the stack. Its saved path will be used in the further ascending step. When descending the tree, the versioned node is maybe found since the insert operation does not use lock-coupling to heighten concurrent execution. That is, the child node could be split by another insert operation after taking the child node pointer from its parent node. In this case, the non-versioned nodes can be found by moving right via the version pointers. The version pointers points to each derived nodes that have been created due to node-split, so that all nodes derived from the original node can be visited by moving right. Before a process moves right, the pushed node is popped as shown in Fig. 8 (line 28). If a process finds non-versioned node leading to geometrically optimal node, it is pushed to the stack. Finally, only non-versioned nodes are in turn pushed to the stack, and if a process reaches to a leaf node, it returns.

If there is space to insert the new key to the found leaf node, it is simply inserted. If region of the node is changed due to the key insertion, the region propagation occurs

to reflect this change to the parent node. The region propagation moves up the tree until the region of ancestor nodes do not need to be changed any more. It may meet a versioned node during the region propagation. However, this case is little different to the wrong path correction of the descending step. The descending step fixes the path by finding a geometrically optimal node among all non-versioned nodes that have been derived from the versioned node. That is, all non-versioned nodes derived from the versioned node should be visited. But the ascending step only moves right via version pointers until the non-versioned node, that contains the pointer to the corresponding child node, is found.

```

1  updateParent(STACK s, NODE n, RECT r)33      distribute p's entries to p1 and p2
2  if(s is not empty)                               34      insert entries(n1, n2) to optimal node
3  p = pop(s); w-lock(p)                             35      set p1 and p2 to p's version pointers
4  if(p is versioned)                               36      mark the version bit of p as true
5  p = findVersion(p, n)                             37      splitNode(s, p, p1, p2)
6  end                                               38      else
7  unlock(n)                                         39      insert entries(n1, n2) to p;
8  e = entry containing n's pointer in p          40      if(MBR(p) has changed)
9  update e with r                                   41      updateParent(s, p, MBR(p))
10 if(MBR(p) is changed)                             42      else
11  updateParent(s, p, MBR(p))                     43      unlock(p)
12 else                                               44      end
13  unlock(p)                                         45      end
14 end                                               46      end
15 else                                               47
16  unlock(n)                                         48      NODE findVersion(NODE p, n)
17 end                                               49      if(p is versioned)
18 end                                               50      unlock(p); w-lock(p.leftVersion)
19                                                    51      p = findVersion(p.leftVersion, n)
20 splitNode(STACK s, NODE n, n1, n2)          52      if(p is not null)
21 if(s is empty)                                     53      return p
22 p = root                                           54      end
23 else                                               55      w-lock(p.rightVersion)
24 p = pop(s); w-lock(p)                             56      return findVersion(p.rightVersion, n)
25 if(p is versioned)                               57      end
26  unlock(p)                                         58      if(p has the entry that contains n)
27  p = findVersion(p, n)                             59      return p
28 end                                               60      end
29  unlock(n)                                         61      unlock(p)
30 if(p is full)                                     62      return null
31  s-lock(p)                                         63      end
32  p1, p2 = create new node

```

Fig. 9. The insert operation (ascending step)

A node is split if there is no space to insert the new key to the leaf node. Unlike the traditional node-split, the splitting process does not physically modify the splitting

node, but rather marks it as “versioned”, then two other nodes are created for compensating its intactness. Finally, the pointers of newly created two nodes are reflected on parent node. If there is no space to insert them in the parent node, the split process moves up to its parent node again.

3.4 Space Reclamation

According to the instant versioning technique, the garbage space, such as versioned entries and versioned nodes, is essentially created. This subsection discusses how to reclaim garbage space and who collects it. In order to implement the timestamp, a logical version number is used.

The versioned entries are not returned to system, but they are reused by internal update operations. Each node keeps a logical entry version number (*levn*) in its header section. When versioning an entry, *levn* is increased and the new value is assigned to the versioned entry. The internal operations memorize *levn* before they visit the node. If *levn* of the versioned entry is smaller or equal to the smallest one among the internal operations that are currently visiting the node, the versioned entry can be reused.

The reclamation of versioned nodes is similar to that of versioned entries. That is, tree globally keeps another logical version number, a logical node version number (*lnvn*), and the external operations memorize *lnvn* before they start. When versioning a node, *lnvn* is increased and the new value is inserted into the collector queue with the node. Garbage collector is activated on a regular basis, and pops a node from the queue. If *lnvn* of the popped node is smaller or equal to the smallest one among the active external operations, it is returned to system.

Actually, the space reclamation does not require heavy overhead since the versioned entries are reused and the versioned nodes are reclaimed by an independent process without disturbing the normal operations.

4 Consistency

This section discusses the phantom problem that is a common requirement of database systems. It is difficult to avoid the phantom problem by index itself. One simple way to avoid the phantom problem is to hold the lock on every node (leaf nodes and non-leaf nodes) that search operations visit until transaction finishes. However the concurrent execution is severely decreased. Rlink-tree introduces a simplified form of predicate locks [3], where exclusive predicates consist of a single rectangle and shared predicates consist of query range. Insert operations check shared predicates with their single rectangle, and if they conflict, they suspend until the conflicted shared predicate is released. The main advantage of the predicate locking is to isolate the concurrency technique on index from the phantom problem. That is, when an operation is passed the predicate locking manager, it can freely access any nodes on the tree without considering the phantom problem. The predicate locking could be employed with R^{ver} -tree. However, it does not utilize the advantage of this paper, i.e. non-blocking search operation, since search operations could be blocked before entering to the index by the predicate locking.

A more effective solution for R^{ver} -tree is to cooperate with the multi-version record manager. Read transaction gets a candidate object qualifying their search condition from the index, and then the candidate object is compared with multiple versions in record manager. If the timestamp of the candidate object is greater than that of read transaction, it is ignored since the candidate object was created after read transaction has installed, and the next candidate object is got from the index. This approach needs to consider a delete operation on index. If a key is deleted after search operation scan the tree, the number of objects got from rescan operation is maybe smaller than that of previous scan. So, the key deletion is not done by the current delete transaction, but it is lately done by a garbage record collector like “ager” in [5]. The ager is an independent process that reclaims garbage records. When reclaiming the garbage records the corresponding keys are deleted from the index.

5 Deadlock

In this section, R^{ver} -tree proves itself to be deadlock-free. Deadlock occurs when there is a cycle of operations waiting for each other. By figuring out locking sequence of sub-functions, it is shown that R^{ver} -tree does not make a cycle.

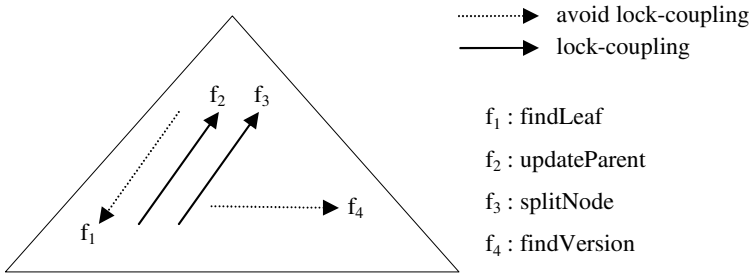


Fig. 10. The locking sequence of sub-functions of the external insert operation

Fig. 11 shows the locking sequence of sub-functions of the external insert operation. Another operations, such as internal operations, external search operation, and external delete operation, are not considered since internal operations do not require locks of two nodes at a time, external search operation does not acquire any lock, and external delete operation uses the same functions in external insert operation. The dot line indicates to release the lock on the visiting node before requiring the lock on next node. And the solid line indicates to use lock-coupling. Deadlock can occurs only when the solid line is crossed. As shown in Fig 11, the solid line is never crossed.

6 Experiments

This section proves the excellence of this technique through comparison to Rlink-tree. It explains experimental environment, and estimates the proposed technique by increasing insert and search process.

6.1 Experimental Environment

Rlink-tree and R^{ver} -tree were implemented in C under GMS [10] which is a spatial database management system. GMS run on Solaris equipped with 8 CPUs of 1.2GHz and main memory of 1G bytes.

The size of node is 4K. The fan-out of non-leaf node and leaf node is each 98 and 81 for Rlink-tree, 89 and 75 for R^{ver} -tree. This experiment does not consider the phantom problem since Rlink-tree does not mention in detail. And the quadratic split algorithm is applied to both indexes.

Initially, the data sets with 10000 rectangles (10 X 10 size) were preloaded in the 20000 X 20000 area. It was equally distributed in total domain area. In actual experiments, each insert process inserts objects with size of 10x10 randomly extracted from total area into the tree. Also, each search process searches with size of 2000x2000 (1% of entire area) randomly extracted. The response time and throughput of search and insert operation is estimated by dynamically increasing the number of insert and search process.

6.2 Experimental Results

Fig. 11 and Fig. 12 are the results of estimating response times and throughputs by increasing number of insert processes to measure insert-workload. It shows that R^{ver} -tree has slightly bad performance even though it tries to improve the concurrent efficiency of the update operations. This is due to the fact that R^{ver} -tree is more split than Rlink-tree since the fan-out of R^{ver} -tree is smaller than Rlink-tree. That is, the overhead caused by correction of wrong path or blocking due to node-split, is increased.

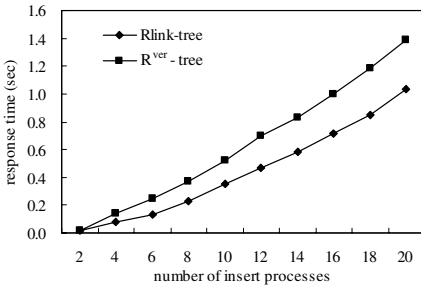


Fig. 11. Response times of insert-workload

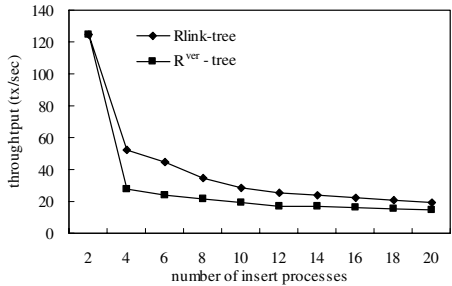


Fig. 12. Throughputs of insert-workload

Fig. 13, Fig. 14, Fig. 15, and Fig. 16 are the results of estimating average response times and throughput of search operations by increasing search operations under the low-contention (4 insert processes) and high-contention (20 insert processes). R^{ver} -tree is showing better performance than Rlink-tree under the low-contention as shown in Fig. 12 and Fig. 13. This is due to the fact that search operations of R^{ver} -tree traverse the tree without any locking. In contrast, those of Rlink-tree are blocked by concurrent insert operations. The difference in performance of both indexes is more severe under the high-contention. Rlink-tree is getting worse as contention is increased. Notice that response times of R^{ver} -tree achieve nearly similar results in two cases,

low-contention and high-contention. Consequently, response times of R^{ver} -tree is shown within expected time even though the contention is increased since search operations of R^{ver} -tree do not require any locking or latching.

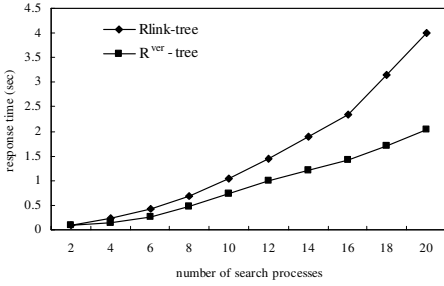


Fig. 13. Response times of search operations (low-contention)

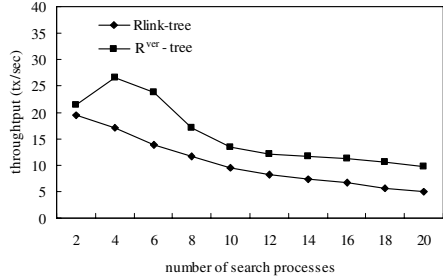


Fig. 14. Throughputs of search operations (low-contention)

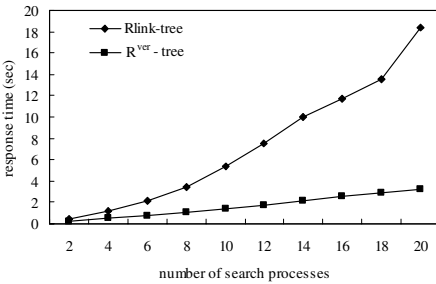


Fig. 15. Response times of search operations (high-contention)

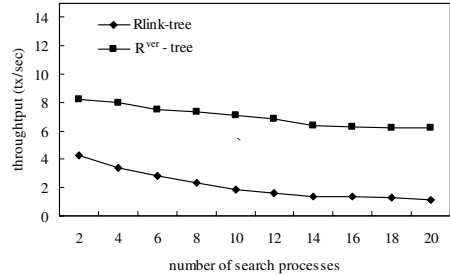


Fig. 16. Throughputs of search operations (high-contention)

7 Conclusions

This paper has designed the modified R-Tree structure with lock-free querying, and implements it on existing spatial database management system. In order to achieve the goal, the instant versioning technique is introduced. This technique has not physically modified data, but rather new version has been created for compensating data intactness. Search operation could access data without any locking or latching by reading old version. This approach has made garbage data due to compensating action, but the reclamation task has not conflicted with the normal operation of index since it has been executed by an independent process. Experiments have shown better performance in search operation at least twice as fast as compared to Rlink-tree.

The further work for this paper is the study of recovery. The reclamation of garbage nodes are executed independently of transactions. If system is crashed after creation of garbage nodes, it is never returned to system even after transaction is committed.

References

1. R. Bayer and M. Schkolnick.: Concurrency of Operations on B-Trees, *Acta Inf.*, Vol. 9, (1977) 1-21
2. J.K. Chen and Y.F. Huang.: A Study of Concurrent Operations on R-Trees, *J. Information Sciences*, Vol 98, (1997) 94-162
3. K. Eswaren, J. Gray, R. Lorie and I. Traiger.: On the Notions of Consistency and Predicate Locks in a Database System, *Comm. ACM*, Vol. 19, No. 11 (11 1976) 624-633
4. A. Guttman.: R-trees: A dynamic index structure for spatial searching, *Proc. ACM SIGMOD Int. Conf. on Management of Data*, (1984) 47-57
5. H.V. Jagadish, Dan Lieuwen, Rajeev Rastogi, Avi Silberschatz, and S. Sudarshan.: Dali: A high performance main-memory storage manager, *Proc. of the Int. Conf. on Very Large Data Bases* (1994)
6. M. Kornacker and D. Banks.: High-Concurrency Locking in R-Trees, *Proc. of the Int. Conf. on Very Large Data Bases*, (9 1995) 134-145
7. M. Kornacker, C. Mohan, and J. Hellerstein.: Concurrency control and recovery in GiST, *Proc. ACM SIGMOD Int. Conf. on Management of Data* (1997)
8. P. Lehman and S. Yao.: Efficient Locking for Concurrent Operations on B-Trees, *ACM TODS*, Vol 6, No. 4 (12 1981)
9. V. Ng and T. Kamada.: Concurrent Accesses to R-Trees, *Proc. Symp. Large Spatial Databases* (1993) 142-161
10. S. Park, W. chung, and M. Kim GMS: Spatial database management system", *Proc. of the KISS Spring Conf* (4 2003)
11. R. Rastogi, S. Seshadri, P. Bohannon, D. Leinbaugh, A. Silberschatz, and S. Sudarshan.: Logical and Physical Versioning in Main Memory Databases, *Proc. of the Int. Conf. on Very Large Data Bases* (8 1997)
12. K.V. Ravi Kanth, Divyakant Agrawal, and Ambuj K. Singh.: Improved concurrency control techniques for multi-dimensional index structures, *Technical Report, Univ. of California at santa Barbara* (1998)
13. A. Prasad Sistla, U. Wolfson, S. Chamberlain, and SonDao.: Modeling and querying moving object, *Proc. of the IEEE Int. Conf. on Data Engineering* (4 1997) 422-432
14. O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang.: Moving objects databases: Issues and solutions, *Proc. of the Int. Conf. on Statistical and Scientific Database Management* (6 1998) 111-122

Importance-Based Web Page Classification Using Cost-Sensitive SVM*

Wei Liu¹, Gui-rong Xue¹, Yong Yu², and Hua-jun Zeng³

¹ Shanghai Jiao Tong University No.800, Dongchuan Road,
Min Hang Shanghai, China 200240
{liuweiwei, grxue}@sjtu.edu.cn

² Shanghai Jiao Tong University Computer Science Department,
Shanghai, China 200030
yyu@cs.sjtu.edu.cn

³ Microsoft Research Asia 5/F, Beijing Sigma Center, No.49, Zhichun Road,
Hai Dian District, Beijing China 100080
hjzeng@microsoft.com

Abstract. Web page classification is facing great challenges since there is a huge repository and diversity of information. As known, each web page varies both in content and quality, just as PageRank suggested. Typical machine learning algorithms take advantage of positive and negative examples to train a classifier; however, it has been neglected that each instance has a different weight, which can be user pre-defined. This paper presents an effective algorithm based on Cost-Sensitive Support Vector Machine (CS-SVM) to improve the accuracy of classification. During the training process of CS-SVM, different cost factors are attached on the training errors to generate an optimized hyperplane. Our experiments show that CS-SVM outperforms SVM on the standard ODP data set. The web pages with relative high PageRank values contribute most to the classifier and using them for training can exceed the random sampling technique.

1 Introduction

The amount of information on the World Wide Web is growing with an incredible speed nowadays. Every day approximate 60 terabytes of new content is added to the Web's 10 billion or so indexable pages[1]. Categorization plays an important role in organizing the web content and people resort to web directories like Yahoo, ODP, and LookSmart etc. for browsing. Web page classification is the task of deciding whether a page belongs to a set of pre-defined category of documents or whether a page is relevant to a certain topic. Initially, classification was done manually by domain experts. But due to the fast growth in online document data, this becomes more difficult with time. Automatic classification schemes can greatly facilitate the process of categorization and many approaches have been proposed, such as K-Nearest Neighbor [2], Bayesian probabilistic models [3], decision trees [4], Support Vector Machine [5] and neural networks [6].

In this paper, we focus on the SVM algorithm, for its wide acceptance due to the solid theoretical basis and high generalization ability. [7] introduced the concepts of

* This work was conducted while the author was doing internship at Microsoft Research Asia.

VC dimension and structural risk minimization, and then described linear SVMs for separable and non-separable data. In classical SVMs, the hard margin loss function is suitable for noise-free data sets. For other general cases, a soft margin loss function is also popularly used, which introduces slack variables to allow some misclassification. Our CS-SVM algorithm is based on the soft margin theory of SVM and deals with the different weights of outliers. SVM can use the kernel trick to map the data into high dimensional or even infinite dimensional space to solve the non-linear problems and here we adopt the linear kernel for simplification.

Many variant forms of SVM have been suggested. [8] introduced Transductive Support Vector Machines (TSVMs) for text classification, which took into account a particular test set and tried to minimize misclassifications of just those particular examples. In [9], Least Square Support Vector Machine (LS-SVM) has been investigated for classification and function estimation. The solution worked with equality instead of inequality constraints and a sum squared error (SSE) cost function. [10] proposed a biased SVM to assign C+ and C- to weight positive and negative errors respectively.

However, previous work on SVM has not realized that each instance in a training data set should be treated differently. Especially in the application of web page classification, most users are concerned about the important or popular pages rather than the infrequently visited web pages or desolate websites. The pages which have many inlinks or authorized inlinks will have high PageRank [11] values. PageRank can be used to characterize the popularity or importance of web pages, relying on the link structure of the web. Therefore, in this paper, we propose the Cost-Sensitive SVM algorithm based on the PageRank importance to improve the classification results.

During the training process of CS-SVM, different cost factors are attached on the training errors to generate an optimized SVM hyperplane. This outperforms the existing technique of the standard SVM. Besides, web pages have different effects on classification – the most contributive samples for classification are not the highest PageRank web pages but the relative high and topic focused ones, thus sampling these most contributive web pages for training is much better than the random sampling technique.

The rest of the paper is organized as follows. In Section 2, the related work is introduced. In Section 3, an overview of SVM is provided. In Section 4, we present the Cost-Sensitive Support Vector Machine algorithm in and the experiment results are reported in Section 5. Finally we draw the conclusions and suggest some future work in Section 6.

2 Related Work

Web Page Classification

In web classification, web pages from one or more sites are assigned to pre-defined categories according to their content. Previous work can be divided into two types:

a) Text content based classification: A collection of keywords and their frequency of the occurrences etc, were calculated from a large collection of text. Then all the documents would be presented as feature vectors and classified into appropriate directories using the KNN, Naïve Bayes, or SVM etc.

b) Link and Tag based classification: Hyperlinks clearly contained high-quality semantic clues that were lost upon a purely text classifier, but exploiting link information was non-trivial because it was noisy. Naive use of terms in the link neighborhood of a document would even hurt the classification performances. [17] put forward a relaxation labeling technique for better classification. Meta data and title were also helpful as described in [18]. [19] used the URLs and table layout for web classification tasks.

However, all the previous work has not considered combining the PageRank values of web pages into classification or taking full use of the importance information on different data. That's what we have explored in our paper.

SVM Variants and Weighted Methods

Transductive Support Vector Machines (TSVMs) [8] for text classification took into account a particular test set and tried to minimize misclassifications of just those particular examples. In [9], a Least Square Support Vector Machine (LS-SVM) was investigated for classification and function estimation. [20] suggested a Laplacian SVM based on manifold regularization, which exploited the geometry of the probability distribution and performed well in handwritten digit recognition and spoken letter recognition. [10] proposed a biased SVM to assign C+ and C- to weight positive and negative errors respectively. When the positive example set was homogenous, i.e. focusing on one topic, and they covered a rather smaller region in the vector space, it tended to extract many negative documents with high precision and produce a more accurate classifier.

There are many weighted methods as well. Weighted least squares regression [21] was sensitive to the effects of outliers, which could be used to maximize the efficiency of parameter estimation. [22] proposed a weighted dissimilarity measure in vectorial spaces to optimize the performance of the nearest neighbor classifier. Some researchers have also combined different features of web pages such as text words, link numbers and the titles etc, with different weights on them to train a classifier or cluster the web documents[19][23].

3 SVM Overview

SVM was first brought forward by Cortes and Vapnik [12] as a learning algorithm for classification and regression. It tried to maximize the margin of confidence of classification on the training data set, which could use the linear, polynomial or radial basis function (RBF) kernels. Now we will outline the main ideas of SVM.

Starting from the simple case of two linearly separable classes, it is assumed that there is a data set of labeled examples: $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, where $y_i \in \{-1, 1\}$. From these training examples the algorithm finds the parameters of the decision function:

$$D(x) = \sum_{i=1}^N w_i \bullet x + b \quad (1)$$

where w_i and b are the adjustable parameters of the decision function. The distance r between the hyperplane and training example x is $|D(x)|/\|w\|$. Supposing that there is a margin M between the hyperplane and supporting vectors, so that

$$\frac{y_k D(x_k)}{\|w\|} \geq M \tag{2}$$

The problem of classification equals to finding the maximum margin of SVM. As in Fig.1, the maximum margin is M^* , P_1, P_2 on the boundary L_1 and Q_1, Q_2 on the boundary L_2 are supporting vectors, which are closest to the hyperplane.

$$M^* = \max M \tag{3}$$

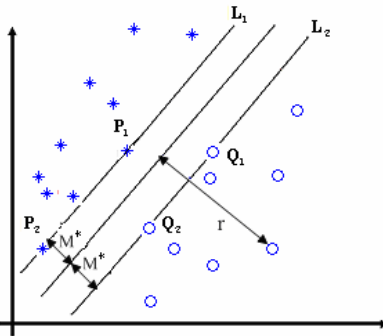


Fig. 1. The linear separable case of Support vector machines, where the positive and negative examples are represented by asterisks and circles respectively

The norm of w can be scaled so that the product of M and $\|w\|$ amounts to 1, which means that the distances of all the training data are at least 1 from the hyperplane.

$$y_i (w \bullet x_i + b) - 1 \geq 0 \quad \forall i \tag{4}$$

Hence, maximizing the margin M is equivalent to minimizing the norm $\|w\|$. It turns out to be a quadratic optimization problem as follows:

$$\min_w \frac{1}{2} \|w\|^2 \quad \text{subject to} \quad y_i D(x_i) \geq 1 \quad \forall i \tag{5}$$

This solution involves contracting a dual problem where a Lagrange multiplier α_k is associated with every constraint in the primary problem:

$$\begin{aligned} \max_{\alpha} Q(\alpha) = \max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j \\ \text{subject to} \quad \sum_i \alpha_i y_i = 0 \quad \text{and} \quad \alpha_i \geq 0 \end{aligned} \tag{6}$$

When the two classes are not linearly separable (e.g. due to noise), the condition (4) can be relaxed by adding the slack variables:

$$y_i (w \bullet x_i + b) \geq 1 - \zeta_i \quad \forall i \tag{7}$$

For minimum error, $\zeta_i \geq 0$ should be minimized as well as $\|w\|$, and the objective function will become:

$$\text{minimize } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^p \zeta_i^k \quad \text{subject to } y_i (w \bullet x_i + b) \geq 1 - \zeta_i \text{ and } \zeta_i \geq 0 \quad \forall i \tag{8}$$

Here C is the trade-off between maximizing the margin and minimizing the train errors. Small-valued C tends to emphasize the margin; otherwise it tends to overfit the training data. In (8), every training example’s error is treated equally with the uniform parameter C , no matter it is the error of a highly important instance or of the less important one. As a matter of fact, we should distinguish different misclassification constraints of the data objects—we never want an essential instance to be wrongly labeled. In the next section, we will introduce our Cost-Sensitive SVM algorithm in detail.

4 Cost-Sensitive SVM

Generally, SVM doesn’t consider any weighted method for its soft margin error, which will result in mistakes on the important data instances. Supposing every training instance has an importance value, there are two classes A and B , as can be seen in Fig. 2, the asterisks belong to class A and the circles belong to class B . Point P is an essential data instance. We assign a weight 0.9 to it and 0.1 to Q . Thus, the point P has the priority to be correctly classified.

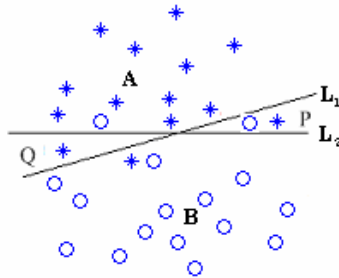


Fig. 2. Different weighted points P and Q in classification, with two hyperplanes L_1 and L_2

The hyperplane L_1 puts P into class B and Q into class A , while L_2 does the contrary. As far as standard SVM concerned, L_1 is the optimal decision boundary, for it only misclassifies three points and the total sum of errors is smaller than that of L_2 . However, if we consider the importance of the data points, it is apparent that L_2 is the better choice as the decision hyperplane.

Therefore, we propose a Cost-Sensitive SVM algorithm to promote the classification accuracy on differently important data. In the soft margin of SVM solution, (8) uses C to balance the margin and training error, which can be modified as:

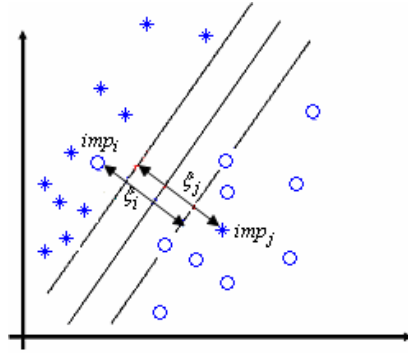


Fig. 3. Different importance on the each data point's error in Cost-Sensitive SVM

$$\min_{\zeta, w, b} \frac{1}{2} \|w\| + \sum_{i=1}^p \text{imp}_i \zeta_i^k \quad \text{subject to } y_i(w \bullet x_i + b) \geq 1 - \zeta_i \quad \zeta_i \geq 0 \quad \forall i \quad (9)$$

Every training data has got a imp_i cost factor for its slack variable, which can be pre-defined by users. For the reason of simplification, we set the power k of the slack variable as 1, and the corresponding Lagrange dual problem of CS-SVM becomes:

$$L(w, b, \zeta, \alpha, \gamma) = \frac{1}{2} \|w\| + \sum_{i=1}^p \text{imp}_i \zeta_i - \sum_{i=1}^p \alpha_i [y_i(w \bullet x_i + b) - 1 + \zeta_i] - \sum_{i=1}^p \gamma_i \zeta_i \quad (10)$$

$$\text{where } \alpha_i \geq 0, \gamma_i \geq 0$$

We can take the partial derivative of L to obtain the dual form of Cost-Sensitive SVM:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^p y_i \alpha_i x_i = 0 \quad (11)$$

$$\frac{\partial L}{\partial \zeta_i} = \text{imp}_i - \alpha_i - \gamma_i = 0 \quad (12)$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^p \alpha_i \gamma_i = 0 \quad (13)$$

Substituting (11), (12) and (13) into primal Lagrange (10), we'll get the dual formulation for 1-norm soft margin problem:

$$L(w, b, \zeta, \alpha, \gamma) = \sum_{i=1}^p \alpha_i - \frac{1}{2} \sum_{i=1}^p \sum_{j=1}^p y_i y_j \alpha_i \alpha_j x_i x_j \quad (14)$$

$$\text{subject to } 0 \leq \alpha_i \leq \text{imp}_i \quad \sum_{i=1}^p \alpha_i y_i = 0 \quad \forall i$$

The Karush-Kuhn-Tucker [13] conditions are

$$\begin{cases} \alpha_i [y_i (w \bullet x_i + b) - 1 + \zeta_i] = 0 \\ \zeta_i (\alpha_i - imp_i) = 0 \end{cases} \quad \forall i \quad (15)$$

That means the slack variable is not zero only when $\alpha_i = imp_i$.

J. Platt [14] suggested a Sequential Minimal Optimization (SMO) for training SVM, which broke the large quadratic programming (QP) optimization problem like (14) into a series of smallest possible QP problems. The SMO algorithm searches through the feasible region of the dual problem and maximizes the objective function. It works by optimizing two α_i 's at a time, with other α_i 's fixed. When SMO applied to CS-SVM, the optimization process would become as:

Optimizing α_1, α_2 from an old set of feasible solution: $\alpha_1^{old}, \alpha_2^{old}, \alpha_3, \dots, \alpha_N$ (initial α_{old} is set to 0), because $\sum_{i=1}^N \alpha_i y_i = 0$, we'll have:

$$\alpha_1 y_1 + \alpha_2 y_2 = \alpha_1^{old} y_1 + \alpha_2^{old} y_2 \quad (16)$$

This confines the optimization to be on a deflective line, as shown in the following figure:

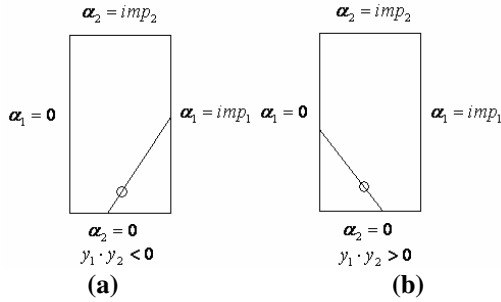


Fig. 4. Optimizing two α_i 's in SMO during a loop

During optimization, there are different upper bounds for α_1 and α_2 according to their importance weights and the Lagrange multipliers will be updated after each loop. In the following two cases, the KKT condition are violated:

$$\begin{cases} \alpha_i < imp_i \text{ and } R_i < 0 \\ \alpha_i > 0 \text{ and } R_i > 0 \end{cases} \quad \text{where } R_i = y_i E_i = y_i (w \bullet x_i - b - y_i) \quad (17)$$

The heuristic for picking two α_i 's for optimization in the SMO is two loops' sweeping, i.e. the outer loop selects the first α_i from the examples that violates the KKT conditions and inner loop looks for a non-boundary example that maximizes $|E_1 - E_2|$. Because in CS-SVM there are different constraints on the two α_i 's,

their adjustment is biased compared with Standard SMO algorithm for SVM. Finally after many sweeps, CS-SVM will reach the optimal solution.

The imp_i 's ($i=0, 1, \dots, p$) can be pre-defined by users before training the classifier. In our experiment, we take the original integral PageRank values for each imp_i .

5 Experiment Evaluation

In this section, we provide the evaluation of CS-SVM algorithm on the 1,546,439 web pages of Open Directory Project. All of the experiments are done in two Intel® Xeon™ CPU 3.06GHZ machines with 3.87 GB of RAM and 2.0GB of RAM respectively. CS-SVM is implemented based on SVM ^{light} [15].

5.1 Data Set

Our experiments are conducted on the 174 classes of ODP's second level directory. In fact, there are totally 388 classes in ODP. We remove the classes whose page numbers are less than 1,000 so that the training and testing data for each class in CS-SVM will be large enough. In this paper, we generate two data sets for following experiments. The first one has three 15,000 web page subsets, each randomly chosen from the remaining 1,485,540 web pages. The second includes eleven groups, each of which has 100 web pages at a specified PageRank value (0~10) and the corresponding test sets are 100 web pages sets randomly selected from rest web pages.

5.2 Evaluation Metrics

We use Precision, Recall and F_1 measures to evaluate the results. As for multiple categories, there are macro-average and micro-average measures [16]. Both of them are used in our experiments.

5.3 Result Analysis

First, we conduct an experiment on the first data set, i.e. three groups, each of which contains 15,000 random web pages from ODP. The results on CS-SVM and standard SVM with ten-fold cross validation are presented in Table 1.

Compared with SVM, the MicroP, MicroR and MicroF1 has improved 6.8%, and MacroP has improved 39%, MacroR 47% and MacroF1 49% respectively.

Table 1. The average results of three subsets. (MicroP = MicroPrecision, MicroR = MicroRecall, MicroP = MicroPrecision, MicroR = MicroRecall).

	CS-SVM	SVM
MicroP	0.410666	0.384457
MicroR	0.410666	0.384457
MicroF1	0.410666	0.384457
MacroP	0.236968	0.170467
MacroR	0.193954	0.131289
MacroF1	0.199911	0.133953

Next, we perform ten experiments to compare the effects of web pages with different PageRank values in a classification task. The 1,485,540 web pages are divided into 90% training set and 10% testing set. Then 100 web pages are selected randomly from original training set for each PageRank value, and 100 web pages from testing set. The result is shown in Fig.5 as below.

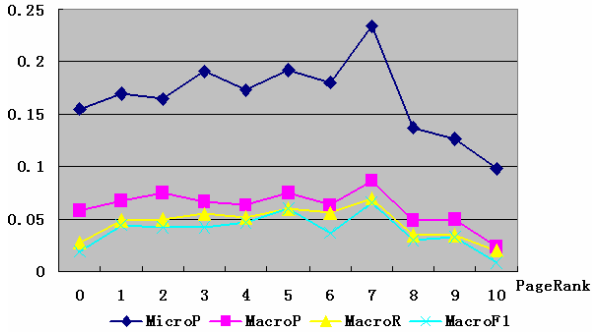


Fig. 5. Effects of web pages with different PageRank values

The web pages whose PageRank is equal to 7 plays the most important role in the classification and surprisingly, the pages whose PageRank values are between 8 and 10 perform worse even than the low PageRank ones. We analyze the corresponding web pages for PageRank 8~10, and find that most of them are homepages of popular web sites or business service lists etc., which contain many outlinks but very short length of paragraphs. For example, the 169055th web page with PageRank 10 only has 69 words, including the mail contact information etc. Therefore, they cannot offer enough information because we use the TFIDF feature vectors for classification. And low PageRank samples often contain many texts and somewhat can reflect the characteristics of the category that they belong to, hence, their performances are not so bad. The most contributive web pages are the ones with PageRank 7, because they are not only of high quality and popular but also contain adequate information.

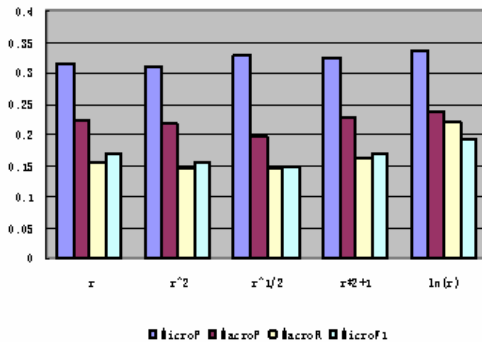


Fig. 6. Five forms of PageRank used in CS-SVM

During our experiments, we make further exploration to find which form of PageRank is the best for CS-SVM. There are squared form, linear form and logarithmic format etc. For both Micro and Macro measures, the logarithmic PageRank is the best choice as shown in Fig. 6

Finally, we apply our importance based classification method to the sampling technique. High PageRank web pages between 5 and 8 in a 15000 random set are sampled for training and the compared same size samples are randomly selected. The ten fold cross validation result is shown in Table 2:

Table 2. Ten cross validation result on random 15000 pages

	<i>Micro Precision</i>
PageRank(5~8)	0.4165871
Random	0.3536129

The web pages with PageRank(5~8) contain the most contributive information and their classification result exceeds the random sampling technique.

6 Conclusions and Future Work

In this paper, we improve the classification performance by using the Cost-Sensitive SVM algorithm, compared with the standard SVM. Considering the web pages have different quality and popularity, we utilize PageRank values as different error requirements in the soft margin of CS-SVM, which results in an optimized decision hyperplane. Experiments show that CS-SVM outperforms SVM in a great extent and the most contributive samples for classification are not the highest PageRank web pages but the relative high PageRank and rich text information providing ones. Besides, the best form of PageRank in CS-SVM is the logarithmic value. Finally, using the web pages whose PageRank values are between 5 and 8 as the training data set can get better performance than the random sampling method.

In future, we can further take advantage of the link structure of web in the various forms and combine them with SVM. Like the Laplacian method in [20], a link graph can be constructed with each edge differently weighted, which need more knowledge on manifold theory. Moreover, the importance based idea can be used in the tasks of web page clustering and other applications, too.

References

1. W.Roush. Search Beyond Google. MIT technology review, pages, (2004) 34-35.
2. Yiming Y., Xin L.: A Reexamination of Text Categorization Methods, In proceedings of the 22th International Conference on Research and Development in Information Retrieval, University of California, Berkeley, USA (1999) 42-49
3. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classification. In Proceedings of AAI-98 Workshop on Learning for Text Categorization, Madison, WI, (1998) 41-48.
4. Lewis, D.D., Ringuette, M.: A Classification of Two Learning Algorithms for Text Categorization. In Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval (1994) 81-93

5. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Proceedings of 10th European Conference on Machine Learning, (1998) 137-142.
6. M.E. Ruiz and P. Srinivasan. Hierarchical text categorization using neural networks. In Information Retrieval, 5(1), (2002) 87-118.
7. Burges, C.: A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, Vol. 2, No. 2, (Jun. 1998) 121-167
8. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In Proceedings of the 16th International Conference on Machine Learning (ICML), Bled, Slovenia, (1999) 200-209.
9. Suykens, J.A.K. Vandewalle, J.: Least Squares Support Vector Machine Classifiers. Neural Processing Letters, 9(3) (1999) 293-300
10. Bing, L., Yang, D., Xiaoli, L., Wee Sum, L.: Building Text Classifiers Using Positive and Unlabeled Examples. In Proceedings of International Conference on Data Mining (2003) 179-186
11. Brin, S., Page, L.: The Anatomy of a Large-scale Hypertextual Web Search Engine. In Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia, (1998)
12. Bernhard E., B., Isabelle M., G., Vladimir N., V.: A Training Algorithm for Optimal Margin Classifiers, In Proceedings of International Conference on Computational Learning Theory, (1992) 144-152
13. Kuhn, H., Tucker, A.: Nonlinear Programming. In Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, (1951) 481-492
14. Platt, J.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines, In Advances in Kernel Methods - Support Vector Learning, (1998) 185-208
15. Joachims, T.: Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, MIT-Press, (1999)
16. Yiming, Y.: An Evaluation of Statistical Approaches to Text Categorization. Information Retrieval, 1(1-2) (1999) 69-90
17. Chakrabarti, S., Dom, B., Indyk, P.: Enhanced Hypertext Categorization Using Hyperlinks. In Proceedings of ACM Special Interest Group on Management of Data, 27(2): (June 1998) 307-318
18. Attardi, G., Gull, A., Sebastiani, F.: Automatic Web Page Categorization by Link and Context Analysis. In Proceedings of 1st European Symposium on Telematics, Hypermedia and Artificial Intelligence, (Varese, IT) (1999) 12
19. L.k. Shih, D.R. Karger.: Using URLs and Table Layout for Web Classification Tasks, In Proceedings of the 13th international conference on World Wide Web (2004)
20. Belkin, M., Niyogi, P., Sindhvani, V.: Manifold Regularization: a Geometric Framework for Learning from Examples, University of Chicago Computer Science Technical Report TR-2004-06, (2004)
21. Carroll, R.J., Ruppert, D.: Transformation and Weighting in Regression, Chapman and Hall, New York, (1998)
22. Paredes, R., Vidal, E.: A Nearest Neighbor Weighted Measure in Classification Problems. In Proceedings of VIII Simposium Nacional de Reconocimiento de Formas y Analisis de Imagenes, volume 1, Bilbao, Spain, (May 1999) 437-444
23. Shen, H., Gui-Rong, X., Yong, Y., Benyu, Z., Zheng, C., Wei-Ying, M.: Multi-type Features based Web Document Clustering. In Proceedings of the 5th International Conference on Web Information Systems Engineering, Nov. 22-24, Brisbane, Australia. (2004)

An Efficient Approach for Interactive Mining of Frequent Itemsets

Zhi-Hong Deng, Xin Li, and Shi-Wei Tang

National Laboratory of Machine Perception,
School of Electronics Engineering and Computer Science,
Peking University, Beijing 100871, China
{zhdeng, lix}@cis.pku.edu.cn, tsw@pku.edu.cn

Abstract. There have been many studies on efficient discovery of frequent itemsets in large databases. However, it is nontrivial to mine frequent itemsets under interactive circumstances where users often change minimum support threshold (minsup) because the change of minsup may invalidate existing frequent itemsets or introduce new frequent itemsets. In this paper, we propose an efficient interactive mining technique based on a novel vertical itemset tree (VI-tree) structure. An important feature of our algorithm is that it does not have to re-examine the existing frequent itemsets when minsup becomes small. Such feature makes it very efficient for interactive mining. The algorithm we proposed has been implemented and its performance is compared with re-running Eclat, a vertical mining algorithm, under different minsup. Experimental results show that our algorithm is over two orders of magnitude faster than the latter in average.

1 Introduction

Data mining has attracted tremendous amount of attention in the database research community due to its wide applicability in many areas. Frequent itemset mining plays an essential role in many important data mining tasks such as associations [1], correlations [2], sequential itemsets [3], partial periodicity [4], classification [5], etc. The problem is formulated as follows: given a large database of transactions, find all frequent itemsets, where a frequent itemset is a set of items that occur in at least a user-specified percentage of the transaction database. Since the first introduction of mining of frequent itemsets in [1], various algorithms [6-10] have been proposed to discover frequent itemsets efficiently. These algorithms can be partitioned into two categories. One utilizes the traditional horizontal transactional database format, and the other utilizes the vertical transactional database format. Apriori [6] and FPgrowth [9] are most important algorithms belonging to the first category and Eclat [10] is one of the most important algorithms belonging to the second category. As mentioned in [11], mining algorithms using the vertical format have shown to be very effective and usually outperform horizontal approaches.

However, most of the previous work has focused on mining frequent itemsets under specified minimum support threshold (or minsup for short) as soon as possible, and very little work has been done on the mining problem where minsup may change. As stated in [12], users are often unsure about their requirements on the minimum support

at first due to the lack of knowledge about the application domains or the outcomes resulting from different threshold settings. Therefore, they may have to re-execute the mining procedure many times with varied thresholds in order to get satisfied results in real-world applications. This simple method of re-execution is clearly inefficient because all the computations done initially for finding the frequent itemsets under old thresholds are wasted. As a result, it is both desirable and imperative to develop effective approaches for interactive mining, which is the problem of mining frequent itemsets in a database under different thresholds. In 2001, an algorithm called Posteriori [12] for interactive mining has been developed. It is based on the Apriori algorithm and needs n passes over the database where n is the size of the maximal frequent itemset. One year later, Remining [13] based on FPgrowth was proposed. Although it avoids the problem of scanning database n times, it must re-examine all existing frequent itemsets that are still frequent under new threshold due to the divide-and-conquer mining strategy of FPgrowth.

In this paper, we present an algorithm to find the new frequent itemsets with minimal re-computation when the minsup is changed. Our algorithm overcomes the shortcomings of Posteriori and Remining by maintaining a vertical itemset tree. The important characteristics of our algorithm are the following:

1. We adopt vertical data layout. That is, the database is organized as a set of columns with each column storing an ordered list of only the transaction identifiers of the transactions in which the item is contained. As a result, computing the support of itemsets is simple and faster with the vertical layout since it involves only the intersections of lists of transaction identifiers (or tidsets for short).
2. We use a novel data structure called vertical itemset tree (or VI-tree for short) for storing frequent itemsets and negative borders. By VI-tree, we can reduce much runtime for mining frequent itemsets when minsup gets small.
3. Our algorithm just needs one scan of the database in initial mining procedure. In succedent mining procedures, no scans of database are needed.

The remaining of the paper is organized as follows. Section 2 gives a detailed problem description. Section 3 introduces the vertical itemset tree and its construction method. Section 4 develops a VI-tree based interactive mining algorithm of frequent itemsets. Section 5 presents our performance study. Section 6 summarizes our study and points out some future research issues.

2 Problem Description

2.1 Mining of Frequent Itemsets

Let $I = \{a_1, a_2, \dots, a_m\}$ be a set of items. Let $DB = \{T_1, T_2, \dots, T_n\}$ be a transaction database, where T_k ($k \in [1..n]$) is a transaction which has a unique identifier and contains a set of items in I . Given an itemset $A (\subseteq I)$, which is a set of items, a transaction T contains A if and only if $A \subseteq T$. The support of A is the number of transactions containing A in DB . An itemset A is a frequent itemset if A 's support is no less than a predefined minsup ξ .

Given a transaction database DB and a minsup ξ , the problem of finding the complete set of frequent itemsets is called the frequent itemsets mining problem.

2.2 Interactive Mining of Frequent Itemsets

Let FI be the set of frequent itemsets in the database DB , and ξ be the minimum support. After users have found some frequent itemsets, they may be unsatisfied with the mining results and want to try out new results with certain changes on the minimum support thresholds, such as from ξ to ξ' . We call mining frequent itemsets under different minsup is interactive mining of frequent itemsets. The essence of the problem of interactive mining is to find the set FI' of frequent itemsets under a new minsup ξ' .

When the minsup is changed, two cases may happen:

1. $\xi < \xi'$: some frequent itemsets in FI will become infrequent under ξ' . Therefore, these frequent itemsets don't belong to FI' .
2. $\xi > \xi'$: all frequent itemsets in FI will still be frequent under ξ' . Therefore, FI is a subset of FI' . At the same time, some itemsets, which don't belong to FI , will become frequent under ξ' and become an element of FI' .

For the first case, the finding of frequent itemsets is simple and intuitive. Just select those frequent itemsets with support no less than ξ' , and put them to FI' . The algorithm can be found in [12]. In the paper, we concentrate on the second case.

3 Vertical Itemset Tree: Design and Construction

3.1 Preliminaries

Because the vertical itemset tree is based on vertical transactional database format, we first introduce vertical data layout as preliminaries in this session.

Table 1. The database SDB (left) and the vertical format of SDB (right)

Transaction (tid)	Items
1	$a c t w$
2	w
3	$a c t w$
4	$a c d w$
5	$c d w$
6	$a c d$

Item	$tids$
C	1, 3, 4, 5,6
W	1, 2, 3, 4, 5
A	1, 3, 4, 6
D	4, 5, 6
T	1, 3

Let $I = \{a_1, a_2, \dots, a_m\}$ be a set of items, $DB = \{T_1, T_2, \dots, T_n\}$ be a transaction database, and $\Gamma = \{id_1, id_2, \dots, id_n\}$ is the set of tids (identifiers) of all transactions in DB , where id_k ($k \in [1..n]$) is the tid of transaction T_k . For convenience, we assume each id_k ($k \in [1..n]$) is an integer. A set $Y \subseteq \Gamma$ is called a tidset. Given an itemset A, A_{tid} , the tidset of A , is defined as follows:

$$A_{tid} = \{ id_j \mid A \subseteq T_j \}.$$

For the sake of discussion, elements in A_{tid} are listed according to ascending order. It is obvious that the support of A is equal to the number of elements in A_{tid} . As a result, the mining of frequent itemsets is translated into the procedure of intersection of itemsets' tidsets. In the vertical layout, the database consists of a list items followed by their tidsets. As an example, consider the database SDB show in the left of Table 1. There are five different items $I = \{a, c, d, t, w\}$ and six transactions $T = \{1, 2, 3, 4, 5, 6\}$. The right of Table 1 illustrates the vertical format of SDB .

3.2 Vertical Itemset Tree

To design a data structure for efficient interactive mining, let's first examine the ways for vertical mining of frequent itemsets. A variety of frequent itemsets mining algorithms have been proposed [10, 11, 14, 15] that use vertical data layout. In this paper, we discuss Eclat.

Let I be the set of items. Define a function $p: 2^I \times N \rightarrow 2^I$ where $p(A, k) = A[1:k]$, the k length prefix of A . 2^I is the power set of I and we assume that elements in $A \subseteq I$ are listed in some order, such as support ascending order or lexicographic order. Define an equivalence relation θ_k on 2^I as follows: $\forall A, B \in 2^I, \theta_k(A, B) \Leftrightarrow p(A, k) = p(B, k)$. That is, two itemsets are in the same class if they share a common k length prefix. θ_k is called a prefix-based equivalence relation [10]. The equivalence relation partitions the set I into disjoint subsets called equivalence classes. Eclat is based on a recursive decomposition of each class into smaller classes induced by the equivalence relation θ_k . Figure 1 shows how the vertical mining of Eclat would proceed from one class to the next using intersections of tidsets of frequent items.

In Figure 1, we assume that the minsup is 4 and the items list as the children of root according to their support descending order. For each frequent node (itemsets) A , from left to right, A intersects with all frequent left siblings of A to get all children of A . For the children of A , which are frequent, we process them as A . By depth-first or breadth-first manner, we can find all frequent itemsets. For example, the tidsets of a ($a_{tid} = 1346$) and of c ($w_{tid} = 13456$) can be intersected to get the tidset for ac ($ac_{tid} = 1346$). In the same way, we can get aw ($aw_{tid} = 134$). We denote $[a] = \{ac, aw\}$, which are the equivalence class with a as common prefix. Although ac is a frequent children of a , it is not used to generate its children (or its equivalence class) because it don't have any left siblings. aw is also not used to generate its children because it is infrequent. For other nodes, we process them as above. Finally, we can get all frequent itemsets. It is obvious that all frequent itemsets (minsup is 4) are in the tree showed by Figure 1. For more information about Eclat, please refer to [10].

Let us examine what we will get if minsup change from 4 to 3. Figure 2 shows the result of vertical mining when the absolute minsup is 3. We can find facts as follows.

1. The tree in Figure 2 is just the extension of the tree in Figure 1. In Figure2, nodes in shadow are new nodes.
2. All new nodes of Figure2 originate from nodes that are infrequent under 4. These infrequent nodes are registered in the left table of Figure 1.

With these observations, in order to facilitate the later mining, we can construct a vertical itemset tree in the first time mining, and then extend the tree in later mining without re-structuring it. The vertical itemset tree can be constructed as follows.

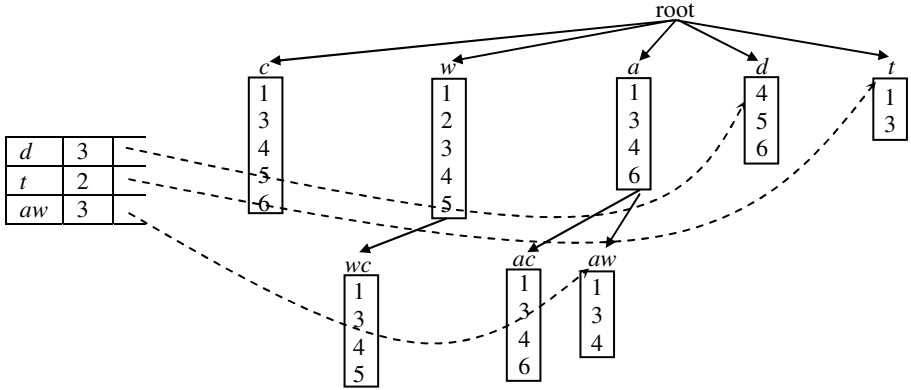


Fig. 1. The VI-tree on SDB (minsup is 4)

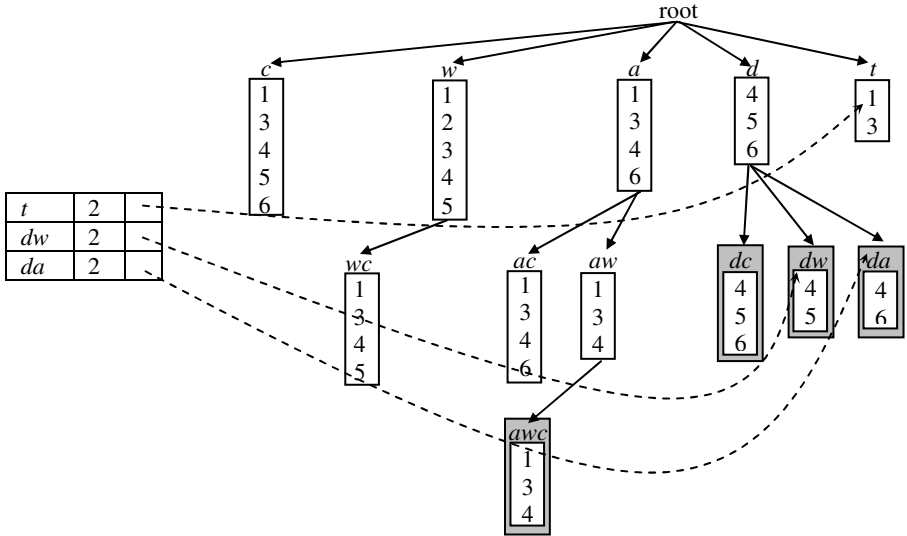


Fig. 2. The VI-tree on SDB (minsup is 3)

First, we change the tree generated by vertical mining with attaching each node with its tidset.

Second, to facilitate finding infrequent itemsets in the tree, a table is built in which each infrequent nodes (itemsets) point to its occurrence in the tree via a head of node-link. For judging whether an infrequent node is frequent or not under new minsup quickly, the support of each infrequent node is also contained in the table. The tree with the associated node-links is shown in Figure 1. If a node in the table is frequent under a new minsup, the node will be deleted from the table and be used to generate all its descendants. For example, because *d* is frequent under 3, new nodes *dc*, *dw*, *da* can be generated as the children of *d*. As a result, we can expand the tree in Figure 1 into the

tree in Figure 2. Of course, the table also should be modified for further mining when the minsup is changed again.

These examples lead to the following design and construction of a vertical itemset tree.

Definition 1 (VI-tree): Given a transaction database DB and a minsup ξ . A vertical itemset tree (or VI-tree for short) is a tree structure defined below.

1. It consists of one *root* labeled as “null”, which lies in level 0.
2. Sort I (the set of all items) in support descending order as OI . Without loss of generalization, Let $OI = \{a_1, a_2, \dots, a_m\}$, while the support of a_i is no less than the support of a_j if i is less than j . a_i with its tidset is inserted into the VI-tree as a children of *root* according to the item order in OI . That is, a_i is a left sibling of a_j if i is less than j . It's obvious that all items are lie in level 1.
3. Given an existing node A , if A is frequent, A will be used to generate its children. The children of A are created as follows: from left to right, for each left sibling of A , B , which is frequent, a node called AB with AB 's tidset is inserted into the tree as a child of A . Let A_1 and A_2 are two children of A . The insertion operation must ensure that A_1 should be the left sibling of A_2 if A_1 was inserted into the tree before A_2 . A simple strategy, in which the newest generated child is always inserted into the tree as the most right child of A , can efficient implements this requirement. In this paper, we adopt this strategy. In addition, items (elements) in AB are listed according to the inverse item order in OI . That is, $AB = a_{i_1}a_{i_2} \dots a_{i_k}$, where $i_x > i_y$ if $x < y$. For example, $AB = a_9a_7a_3$. If A is infrequent, A will not be used to generate its children and will be inserted into a negative border table with its position in the VI-tree.
4. Each node has two fields: itemset-name and tidset, where itemset-name registers which itemset this node represents, tidset registers the tidset of the itemset represented by this node. Of course, the tidset of root is null. A node is call frequent if the number of tid in its tidset is no less than ξ . Otherwise, we call it infrequent.
5. A negative border table (or NB_table for short) is attached with the tree. A node is called negative border if it is infrequent. Each entry (tuple or record) in the NB_table consists of three fields: (1) node-name, (2) the support of node, and (3) head of node-link, which points to the position of the node in the tree. In addition, the entries in NB_table are listed according to the level of node in the tree. The entries of high-level nodes lie above the entries of low-level nodes. Actually, the NB_table consists of l buckets, which are labeled as 1_bucket, 2_bucket, ..., l _bucket. An entry is in i _bucket if the node that it contains lies in i th level of the tree. In the NB_table, entries in i _bucket lie above entries in j _bucket if i is less than j .

According to the definition of a VI-tree, we have some properties as follows.

Property 1: Let $A = a_{i_1}a_{i_2} \dots a_{i_k}$ be a node in the VI-tree. If B is a child of A , B must be in the form of $a_{i_1}a_{i_2} \dots a_{i_k}a_x$, where $x < i_j$ for all j ($1 \leq j \leq k$).

Rational. Refer to 2 and 3 in the definition of a VI-tree.

Property 2: Let A and B be two nodes in the VI-tree. If $A = a_{i_1}a_{i_2} \dots a_{i_k}$ and $B = a_{i_1}a_{i_2} \dots a_{i_k}a_x$, B must be a child of A .

Rational. Let $C = a_{j_1}a_{j_2}\dots a_{j_m}$ be the father of B . According to Property 1, B should be $a_{j_1}a_{j_2}\dots a_{j_m}a_y$. As a result, we have $a_{i_1}a_{i_2}\dots a_{i_k}a_x = a_{j_1}a_{j_2}\dots a_{j_m}a_y$. Because items in a node are ordered according to 3 of the definition, we have $k = m$, $a_{i_s} = a_{j_s}$ ($1 \leq s \leq k$), and $a_x = a_y$. Therefore, we have $C = A$.

Property 3: all frequent itemsets are in the VI-tree.

Rational. According to the definition, the procedure of the VI-tree actually enumerates all frequent itemsets by breadth-first or deep-first manner.

Based on this definition, we have the following VI-tree construction algorithm, which adopt deep-first manner. It should be note that all frequent itemsets are also found as byproduct in our algorithm.

Algorithm VIC (VI-tree construction)

Input: A transaction database DB and an initial minsup ξ .

Output: Its vertical itemset tree (VIT) and the set of all frequent itemsets (FP).

Method:

1. Create the root of a VI-tree, VIT , and label it as rt . Attach a null NB_table to VIT . Set $FP = \emptyset$.
2. Scan the transaction database DB once. Collect the set of items I and their tidsets. Sort I in support descending order as L , the list of items.
3. For each item a in L , do
 - insert a with its tidset into VIT as a child of rt ;
 - if $|a.itemset| \geq \xi$ (that is, a is frequent) then $\{FP = FP \cup \{a\}\}$; else { insert a , with its support and position in VIT , into l_bucket of NB_table; }
4. Call TBD-growth(rt, ξ).

Procedure TBD-growth(F_node, ξ)

```

for A, A ∈ F_node.children ∧ |A.tidset| ≥ ξ , do
  for each B , B ∈ A.left_siblings ∧ |B.tidset| ≥ ξ, do
    R = A ∪ B;
    R.tidset = A.tidset ∩ B.tidset;
    insert R with R.tidset into VIT as a child of A;
    if | R.tidset | ≥ ξ then {FP = FP ∪ {R}\}; else {
      i = level of R in VI-tree;
      insert R, with its support and position in
      VIT, into i_bucket of NB_table; }
  if A has more than one child that is in FP, then { call
  TBD-growth(A, ξ); }

```

4 Interactive Mining of Frequent Itemset with VI-Tree

In section 3.2, we have informally discussed how to expand a VI-tree when minsup become small without re-structuring a new VI-tree under new minsup. In this section, we will study the equivalence of the expanding VI-tree and the new VI-tree. Based on this study, we will propose an efficient algorithm for interacting mining of frequent itemsets.

Let I be a set of items and DB be a transaction database. ξ and ξ' are two minsup and ξ' is less than ξ . We denote that T is the VI-tree constructing on DB with ξ and T' is the VI-tree constructing on DB with ξ' . Definition 2 gives the procedure for expanding T when the minsup is changed from ξ to ξ' .

Definition 2 (expanding rule): For each node A in the NB_table of T , A is delete from the NB_table and is used to generate all of its descendants if its support is no less than ξ' . The process of generating A 's descendants is showed as follows:

1. from left to right, for each left sibling of A , B , which is frequent under ξ' , a node called AB with AB 's tidset is inserted into T as a child of A . If the support of AB is less than ξ' , AB , with its support and position in T , will also be inserted into the i _bucket of the NB_table, where i is the level of AB in T .
2. after generate all children of A , all no-child descendants of A will be generated By calling TBD-growth(A , ξ').

It should be noted that the processing order of nodes in the NB_table is in the descending order. That is, A is processed before B if A lies above B in the NB_table. Let T_1 be the extension of T under minsup ξ' . It is obvious that T_1 also have Property 1 and Property 2. Now, we show that T_1 also have Property 3.

Given a VI-tree VT , let the set of nodes in VT be $VT.Nds$. For each node X in VT , let the set of the children of X be $X_{VT}.children$. We have following Lemma.

Lemma 1: T_1 and T' have the relations as follows:

- 1) $A (\subseteq I)$, if $A \in T_1.Nds$, then $A \in T'.Nds$, and vice versa.
- 2) $\forall A, A_1 \in T_1.Nds$, if $A_1 \in A_{T_1}.children$, then $A_1 \in A_{T'}.children$, and vice versa.

Proof. For convenience, Let $I = \{a_1, a_2, \dots, a_m\}$. Without loss of generalization, we assume that a_1, a_2, \dots, a_m are listed in the descending support order. That is, the support a_i of in DB is no less than that of a_j if i is less than j . In addition, Let the support of an itemsets X be $X.sup$. We first proof 1), and then proof 2).

The Proofing of 1):

If A is a 1-itemset, we have $A \in T_1.Nds$ and $A \in T'$ according to the definition of a VI-tree. In this case, 1) is right. Now, we consider the case that A is a k -itemset ($k > 1$). Let $A = a_{i_k} \dots a_{i_2} a_{i_1}$, where $i_x < i_y$ if $x < y$.

$\Rightarrow A \in T_1.Nds$. there are two cases: $A \in T.Nds$ or $A \in (T_1.Nds - T.Nds)$. If $A \in T.Nds$, we have $a_{i_k} \dots a_{i_3} a_{i_2}.sup \geq \xi$ and $a_{i_k} \dots a_{i_3} a_{i_1}.sup \geq \xi$ according to the construction of T . If $A \in (T_1.Nds - T.Nds)$, then $a_{i_k} \dots a_{i_3} a_{i_2}.sup \geq \xi'$ and $a_{i_k} \dots a_{i_3} a_{i_1}.sup \geq \xi'$ according to the expanding rule. As a result, we have $a_{i_k} \dots a_{i_3} a_{i_2}.sup \geq \xi'$ and $a_{i_k} \dots a_{i_3} a_{i_1}.sup \geq \xi'$ because of $\xi \geq \xi'$. According to the construction of T' , $a_{i_k} \dots a_{i_3} a_{i_2}$ and $a_{i_k} \dots a_{i_3} a_{i_1}$ must be frequent

nodes in T' , and they must be the children of $a_{ik} \dots a_{i3}$. Hence, as a child of $a_{ik} \dots a_{i3} a_{i2}$, $a_{ik} \dots a_{i2} a_{i1}$ must be generated in the processing of constructing T' . That is, $A \in T'.\text{Nds}$.

$\Leftarrow A \in T'.\text{Nds}$. According to the construction of T' , we know $a_{ij}.\text{sup} \geq \xi'$ ($1 \leq j \leq k$), $a_{ik} \dots a_{i3} a_{i2}.\text{sup} \geq \xi'$, and $a_{ik} \dots a_{i3} a_{i1}.\text{sup} \geq \xi'$. Let us see a_{ik} in T_1 . According to the construction of T_1 , $a_{ik} a_{i(k-1)}$, ..., $a_{ik} a_{i2}$, and $a_{ik} a_{i1}$ must be in T_1 as the children of a_{ik} . According to the rule¹: the support of any no-null subset of an itemset X is no less than the support of X , we know $a_{ik} a_{ij}.\text{sup} \geq \xi'$ ($1 \leq j \leq k-1$). From $a_{ik} a_{i(k-1)}$, we know $a_{ik} a_{i(k-1)} a_{ij} \in (a_{ik} a_{i(k-1)})_{T_1}.\text{children} \wedge a_{ik} a_{i(k-1)} a_{ij} \in T_1.\text{Nds} \wedge a_{ik} a_{i(k-1)} a_{ij}.\text{sup} \geq \xi'$ ($1 \leq j \leq k-2$). Iterating above procedure, we know $a_{ik} \dots a_{i3} a_{i2}$ and $a_{ik} \dots a_{i3} a_{i1}$ is nodes in T_1 . Because $a_{ik} \dots a_{i3} a_{i2}.\text{sup} \geq \xi'$ and $a_{ik} \dots a_{i3} a_{i1}.\text{sup} \geq \xi'$ is known, $a_{ik} \dots a_{i2} a_{i1}$, the child of $a_{ik} \dots a_{i3} a_{i2}$ should be in T_1 . That is, $A \in T_1.\text{Nds}$.

The Proofing of 2):

$\Rightarrow A \in T_1.\text{Nds}$, $A_1 \in T_1.\text{Nds}$, and $A_1 \in A_{T_1}.\text{children}$. Let $A = a_{i1} a_{i2} \dots a_{ik}$. According to Property 1, we know $A_1 = a_{i1} a_{i2} \dots a_{ik} a_x$. According to 1), we have $A \in T'.\text{Nds}$ and $A_1 \in T'.\text{Nds}$. As a result, According to Property 2, we have $A_1 \in A_{T'}.\text{children}$.

$\Leftarrow A \in T'.\text{Nds}$, $A_1 \in T'.\text{Nds}$, and $A_1 \in A_{T'}.\text{children}$. In the same way, we can proof $A_1 \in A_{T_1}.\text{children}$.

In fact, Lemma 1 shows that T_1 and T' are one of the same in term of the structure of trees.

Corollary 1: T_1 contains all frequent itemsets under ξ' .

Rational. According to Property 1, T' contains all frequent itemsets under ξ' . As a result, T_1 also contains all frequent itemsets under ξ' because T_1 and T' have the same nodes.

Because itemsets that are frequent with minsup ξ are also frequent with minsup ξ' , we should only find new frequent itemsets² in order to get all frequent itemsets when the minsup is changed from ξ to ξ' . It's very lucky that we can do it in the process of expanding T into T_1 according to definition 2. The following Algorithm gives the details.

Algorithm VIBIM (VI-tree Based Interactive Mining)

Input: VIT and FP under old minsup ξ , and new minsup $\xi' < \xi$;

Output: modified FP and modified VIT under ξ' .

Methods:

$$FP_{new} = \emptyset;$$

```

for  $i$  from 1 to the number of buckets in NB_table do
  for each negative border  $NB$  in  $i\_bucket$  of NB_table do
    if  $|NB.tidset| \geq \xi'$  then {

```

¹ It is obvious that any transaction containing X must contain any no-null subset of X .

² Their supports are less than ξ , but no less than ξ' .

```

delete the entry containing  $NB$  from  $i\_bucket$ 
of  $NB\_table$ ;

 $FP_{new} = FP_{new} \cup \{NB\}$ ;

for each  $B$ ,  $B \in NB.left\_siblings \wedge |B.tidset| \geq \xi'$ , do

     $R = NB \cup B$ ;

     $R.tidset = NB.tidset \cap B.tidset$  ;

    insert  $R$  with  $R.tidset$  into  $VIT$  as a
    child of  $NB$ ;

    if  $|R.tidset| \geq \xi'$  then  $\{FP_{new} = FP_{new} \cup \{R\}\}$ ; else {

         $j = \text{level of } R \text{ in } VIT$ ;

        insert  $R$ , with its support and
        position in  $VIT$ , into  $j\_bucket$ 
        of  $NB\_table$ ;}

    if  $NB$  has more than one child that is in  $FP_{new}$ 
    then{ call  $TBD\text{-}growth(NB, \xi')$ ;}

 $FP = FP \cup FP_{new}$ ;

```

5 Experimental Evaluation

In this section, we present a performance comparison of our approach (VIC+VIBIM) for interactive mining with the re-executing Eclat under different minsup. Our approach runs VIC for the first time mining of frequent itemsets, and then run VIBIM for the later mining procedure under small minsup. It should be pointed out again that we just consider the case where the new minsup is smaller than the old minsup. For the case, where the new minsup is greater than the old minsup, The procedure of mining frequent itemsets is trivial because we can just select those old frequent itemsets with support no less than new minsup in order to get all new frequent itemsets.

All the experiments are performed on a 1.4-GHz Pentium notebook PC machine with 512 megabytes main memory, running on Microsoft Windows XP. All programs are written in Microsoft Visual C++ 6.0. Notice we do not directly compare our absolute number of runtime with those in [10] because different programs may differ on the absolute runtime for the same algorithms. Instead, we implement Eclat in [10] in our environment. The synthetic data sets, which we used for our experiments, were generated using the procedure described in [6]. We report experimental results on two data sets $DB_1 = N1K.T10.L2K.I4.D10K$ and $DB_2 = N1K.T10.L2K.I4.D100K$, where N denotes the number of items, T denotes the average transaction size, L denotes the number of maximal potentially frequent itemsets, I denotes the average frequent itemset size, and D denotes the number of transactions.

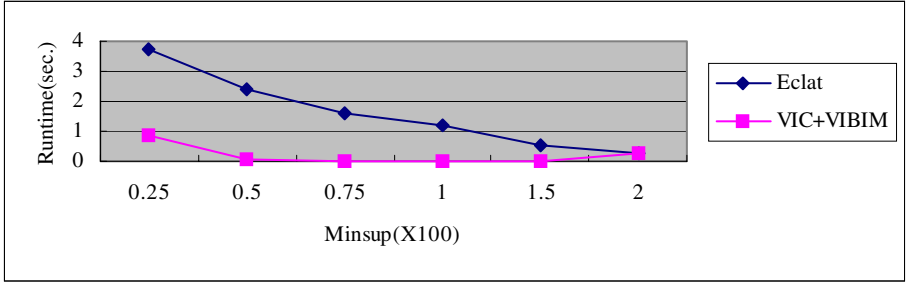


Fig. 3. Runtime under different minimum support thresholds for DB_1

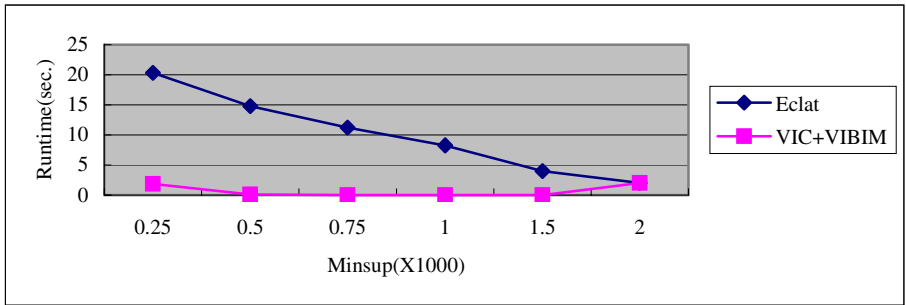


Fig. 4. Runtime under different minimum support thresholds for DB_2

Let us consider the case $\xi' < \xi$, the runtime of Eclat and VIC+VIBIM for DB_1 and DB_2 are plotted in Figure 3 and Figure 4 respectively. We start the mining with relative threshold 2%, and then tune the threshold to 1.5%, 1%, 0.75%, 0.5%, and 0.25% in turn. By multiplying relative threshold by D (the number of transactions), It is easy to get the Minsup in Figure 3 and Figure 4. For both datasets, it is obvious that VIC+VIBIM is much faster than Eclat except for the first time mining. Because VIC, which is used at the first time mining in our approach, is equal to Eclat in terms of the process of mining except that VIC constructs a VI-tree in the first time mining. Figure 3 shows that VIC+VIBIM is 4 to over 1000 times faster than Eclat, and over 200 times in average. Figure 4 shows that VIC+VIBIM is 10 to over 1000 times faster than Eclat, and over 400 in average. All these figures show that our approach outperforms algorithms proposed in [12, 13], where the speedup ratio is between 2 and 6.

6 Conclusions

We studied an efficient approach for interactive mining of frequent patterns. Our approach has characteristics as below. First, by generating and maintaining a TV-tree, the developed technique can fast find all new frequent itemsets under new minsup without re-examining old frequent itemsets. Second, no scanning of databases are needed except for the first time mining. These characteristics make it more suitable for interactive mining than others existing algorithms.

Recently, there have been some interesting studies at mining maximal frequent itemsets [16, 17] and closed frequent itemsets [18, 19]. The extension of our technique for interactive mining of these special frequent itemsets is an interesting topic for future research. In addition, we also take efforts towards space-preserving interactive mining.

Acknowledgement. This research is supported by the National Natural Science Foundation of China under grant No. 60473072. We are also grateful to anonymous reviewers for their comments.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Set of Items in Large Databases. In SIGMOD'93, pp. 207-216.
2. S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. In SIGMOD'97, pp. 265-276.
3. R. Agrawal and R. Srikant. Mining sequential patterns. In ICDE'95, pp. 3-14.
4. J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. In ICDE'99, pp. 106-115.
5. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In KDD'98, pp. 80-86.
6. R. Agrawal and R. Srikant. Fast algorithm for mining Association rules. In VLDB'94, pp. 487-499.
7. J. S. Park, M. S. Chen, and P. S. Yu. An effective hash based algorithm for mining association rules. In SIGMOD'95, pp. 175-186.
8. S. Brin, R. Motwani, J. Ullman, and S. Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In SIGMOD'97, pp. 255-264.
9. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In SIGMOD'00, pp. 1-12.
10. M. J. Zaki. Scalable algorithms for association mining. IEEE TKDE, 12(3): 372-390, 2000.
11. M. Zaki and K. Gouda. Fast vertical mining using diffsets. In SIGKDD'03, pp. 326-335.
12. J. Liu and J. Yin. Towards efficient data re-mining (DRM). In PAKDD'01, pp. 406-412.
13. X. L. Ma, S. W. Tang, D. Q. Yang, and X. P. Du. Towards Efficient Re-mining of Frequent Patterns upon Threshold Changes. In WAIM'02, pp. 80-91.
14. B. Dunkel and N. Soparkar. Data Organization and Access for Efficient Data Mining. In ICDE'99 pp. 522-529.
15. P. Shenoy, J. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa, and D. Shah. Turbo Charging Vertical Mining of Large Databases. In SIGMOD'00, pp. 22-33.
16. R. J. Bayardo. Efficiently mining long patterns from databases. In SIGMOD'98, pp. 85-93.
17. D. Burdick, M. Calimlim, and J. Gehrke. MAFIA: A maximal frequent itemset algorithm for transactional databases. In ICDE'01, pp. 443-452.
18. M. Zaki and C. Hsiao. CHARM: An efficient algorithm for closed itemset mining. In SDM'02, pp. 12-28.
19. J. Y. Wang, J. Han, and J. Pei. CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In SIGKDD'03, pp. 236-245.

Similarity Search with Implicit Object Features

Yi Luo^{1,2}, Zheng Liu¹, Xuemin Lin^{1,2}, Wei Wang¹, and Jeffrey Xu Yu³

¹ The University of News South Wales, Sydney, Australia
{luoyi, zliu, lxue, weiw}@cse.unsw.edu.au

² National ICT Australia, Sydney, Australia

³ The Chinese University of Hong Kong, Hong Kong, China
yu@cse.cuhk.edu.hk

Abstract. Driven by many real applications, in this paper we study the problem of similarity search with implicit object features; that is, the features of each object are not pre-computed/evaluated. As the existing similarity search techniques are not applicable, a novel and efficient algorithm is developed in this paper to approach the problem. The R-tree based algorithm consists of two steps: feature evaluation and similarity search. Our performance evaluation demonstrates that the algorithm is very efficient for large spatial datasets.

1 Introduction

Similarity search is fundamental to many applications involving spatial data analysis. Many research results [1,4,6,8,7,10] have been published in the last decade, where the most popular similarity model is based on a feature vector for each data object. In such a model, each data object, available for similarity search, is represented as a vector, and the similarity between objects is measured by the distance between the vectors. Such applications include image similarity retrieval [4,10], shape similarity search [6,8] and similarity search on spatio-temporal trajectories [1,7]. The k -nearest neighbor (KNN) search is one of the most important similarity search queries. For a query object q and a query parameter k , KNN is to find the k objects that are most similar to q [5,11].

Consider that in many applications, objects for similarity search are not pre-defined; consequently, the feature vector for each object is not pre-computed and stored in a database. For instance, ornithologists may want to identify similar bird communities for selecting a future research target or for behavior predication. A cluster of bird nests is an object. In the application, nest positions are changing regularly and definition of a cluster may vary from time to time because of difference research orientation. Feature groups are represented as groups of polygons. For example, the open water map is a feature group, including lakes, rivers and springs as polygons. Other feature groups are the vegetation map including forests of specific vegetation, the predator distribution map including communities of predatory birds, and man-made structure map including towns, high ways and villages. Moreover, maps of rainfall precipitation and temperature should also be considered; but in these contour maps, each value range could

correspond to a feature group. In the application, a cluster of bird nests can be evaluated based on the distances to the nearest feature in each feature group, such as the nearest open water place and the nearest town. Figure 1 illustrates a cluster of nests and a nearby lake represented as a feature polygon.

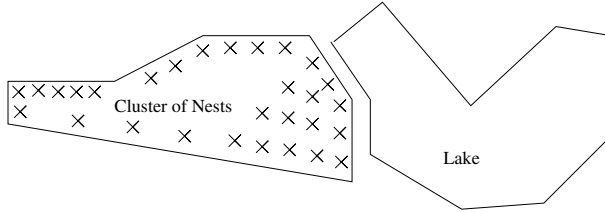


Fig. 1. Ornithology Study

Similar applications lie in road traffic analysis, urban development, crime analysis, etc.

Motivated by the above applications, in this paper we study the problem of a non-conventional KNN, where the feature vector of an object is not pre-computed, namely SSIOF(Similarity Search with Implicit Object Features). In particular, we study the KNN problem where each object is a set of points in 2-dimensional space, and each object is evaluated against d groups of features to obtain a d -dimensional feature vector. By effectively characterizing the results' properties, we develop an efficient and novel R -tree based algorithm to evaluate features of each object. Then, an effective filtering technique is developed to prune away objects (clusters) as many as possible before a precise computation. These are the contributions of the paper. Our performance study demonstrates that our techniques are very efficient to process large spatial datasets.

The remaining paper is organized as follows. Section 2 presents the preliminaries. Section 3 and 4 presents our algorithms and the analysis. Experiment results are reported in Section 5. Finally, Section 6 concludes the paper.

2 Preliminaries

In this section, we start with formally defining the problem and then introduce some necessary background.

2.1 Statement

In a 2-dimensional space, given n clusters C_1, C_2, \dots, C_n and d categories/groups of features $\pi_1, \pi_2, \dots, \pi_d$. Each cluster C_i is a set of points and each feature is a polygon. We use F_j to denote a feature and pt as a point.

Suppose the distance between a cluster C_i and a feature(polygon) F_j , denoted as $d(C_i, F_j)$, is defined as the average Euclidean distance from each point pt in

C to the polygon. Here, the distance between a point and a feature $dist(pt, F)$ is the minimum distance between the point and the edges of the feature. The aggregational *feature evaluation* of a cluster C_i with respect to a feature category π_k is the distance from C_i to its nearest polygon in π_k , denoted as $\phi(C_i, \pi_k)$. The problem of Similarity Search in Implicit Feature Space (SSIOF) is to find k most similar clusters to the given cluster C_0 based on the following similarity measure:

$$Sim(C_i, C_0) = \|(\phi(C_i, \pi_1), \dots, \phi(C_i, \pi_d)), (\phi(C_0, \pi_1), \dots, \phi(C_0, \pi_d))\|_f \quad (1)$$

$\|\cdot\|_f$ is Euclidean or Manhattan distance function; we use the Manhattan distance in our paper.

2.2 R-Tree Index

R-tree is a widely used index for spatial objects based on B^+ -trees, which organizes geometric objects by recursively grouping neighboring objects and representing them by minimum bounding rectangles(MBRs). A node of R-tree corresponds to a disk page. An intermediate node maintains a set of MBRs and pointers which represent the children nodes, while a leaf node contains a set of spatial objects with their positions in the database. Fig. 2 shows an instance of R-tree.

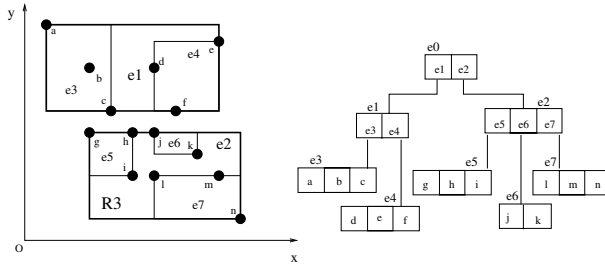


Fig. 2. R-Tree Example

In this paper, we choose one of the most popular variations R^* -tree to index each feature categories and perform our evaluations. Each polygon is represented by its MBR first, then those MBRs is indexed by R^* -tree.

3 Evaluation and Search (ES) Algorithm

Our proposed algorithm *ES* for solving the SSIOF problem contains two major steps:

- 1) **Feature Evaluation:** in this step, we try to find all possible feature candidates for each pair of cluster and feature group.
- 2) **Similarity Search:** this step is to compute the k most similar clusters to the query C_0 , based on the candidates outputted in the previous step.

3.1 Feature Evaluation

Let N_{C_i} be the MBR of a cluster C_i with 4 edges r_1, r_2, r_3 and r_4 ; and N_{F_j} be the MBR of a feature polygon F_j with 4 edges s_1, s_2, s_3 and s_4 . We assume that N_{C_i} and N_{F_j} do not overlap. We will first define some useful metrics between MBR's for later discussion.

$L(r_k, s_l)$ denotes the minimum distance between two points falling on r_k and s_l , and $U(r_k, s_l)$ denotes the maximum distance between two points falling on r_k and s_l [2]. Thus the minimum of distance between two points contained in N_{C_i} and N_{F_j} can be expressed as:

$$\min L(N_{C_i}, N_{F_j}) = \min\{L(r_k, s_l)\} \tag{2}$$

Similarly, we have:

$$\min U(N_{C_i}, N_{F_j}) = \min\{U(r_k, s_l)\} \tag{3}$$

We also define the following distance. For the cluster C and polygon F ,

$$\max \min U(N_{C_i}, N_{F_j}) = \max_k \{ \min_l \{ U(r_k, s_l) \} \} \tag{4}$$

Figure 3 shows the different metrics.

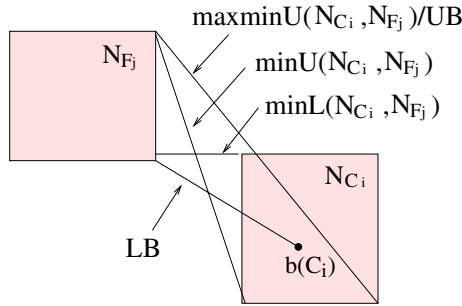


Fig. 3. Distance Matrices

Pruning with the Lower and Upper Bounds

In the SSIOFproblem, the similarity is not measured between points but clusters and polygons, so it's too expensive to compute precise distances on pairs of clusters and features, which makes it necessary to use relatively tight lower and upper bounds for pruning. Then precise distances could be computed only on a small set of clusters and features.

Consider a cluster C_i bounded by MBR N_{C_i} and a feature F_j in MBR N_{F_j} , the lower and upper bounds of the distance between these two are :

$$d_{LB}(C_i, F_j) = L(b(C_i), N_{F_j}) \tag{5}$$

and

$$d_{UB}(C_i, F_j) = \max_{min} U(N_{C_i}, N_{F_j}). \quad (6)$$

$b(C_i)$ in the Equation 5 is the centroid of the cluster C_i computed by the average coordinates of all points in C_i on each dimension. We use $L(b(C_i), N_{F_j})$ to denote the minimal distance between $b(C_i)$ and a point in rectangle N_{F_j} . The lower and upper bounds are illustrated in Figure 3 as LB and UB respectively.

The correctness of lower bound is proved in [9]. By definition, $d(C, F)$ is the average of $dist(pt, F)$ for all $pt \in C$. Based on the inequality:

$$\sum_{i=1}^K \sqrt{x_i^2 + y_i^2} \geq \sqrt{\left(\sum_{i=1}^K x_i\right)^2 + \left(\sum_{i=1}^K y_i\right)^2}$$

$d(C, F)$ is no less than the distance from $b(C)$ to some points inside N_C , which is no less than $L(b(C), N_F)$. Lemma 1 shows the correctness of the upper bounds.

Lemma 1. *For a cluster C_i in MBR N_{C_i} and a feature F_j in MBR N_{F_j} , the upper bound of $d(C_i, F_j)$ is $\max_{min} U(N_{C_i}, N_{F_j})$.*

Proof. Suppose that N_{F_j} is bounded by s_l ($l = 1..4$). Since N_{F_j} is the minimal bound rectangle of F_j , there must be at least a point of F_j on each s_l . Thus the upper bound of $dist(pt, F_j)$ equals $\min_l U(pt, s_l)$. Consider all points on N_{C_i} , the upper bound of $d(C_i, F_j)$ is $\max_{pt \in C_i} dist(pt, F_j)$, which is no larger than $\max_{min} U(N_{C_i}, N_{F_j})$. \square

When N_{F_j} and N_{C_i} overlaps, it can be immediately verified that the above bounds hold. When a feature group is indexed by an R -tree, the lemma still holds if we change N_{F_j} to the MBR of an R -tree node. This gives us the opportunity to prune features while traversing the index.

R-Tree Based Pruning. Making use of the index on each feature group could speed up the process of feature evaluation. Next we will introduce the pruning technique for a feature group π_k indexed by an R -tree T_{π_k} , as shown in Algorithm 1. Each node of T_{π_k} corresponds to a disk page. To lower the disk I/O cost, we traverse T_{π_k} using the following strategy which allow us to visit each R -tree node at most once.

The goal is to find a set of candidate features for each cluster. For each cluster C_i , we maintain a candidate list $L(C_i, \pi_k)$, implemented as a heap. Each list entry e is either the MBR of a non-leaf R -tree node or the MBR of a feature polygon, corresponding the intermediate levels and the leaf level in the R -tree. As mentioned above, the lower and upper bounds hold on both kinds of MBRs, denoted as $d_{LB}(C_i, e)$ and $d_{UB}(C_i, e)$. For any pair of entries in the list, their bounds overlap. $\phi_{LB}(C_i, \pi_k)$, $\phi_{UB}(C_i, \pi_k)$ and $q(C_i, \pi_k)$ are used to record the minimum of $d_{LB}(C_i, e)$, the minimum of $d_{UB}(C_i, e)$ and the maximum of $d_{LB}(C_i, e)$ for each list, respectively. $\phi_{LB}(C_i, \pi_k) = \phi_{UB}(C_i, \pi_k) = \infty$ and $q(C_i, \pi_k) = 0$ initially.

Algorithm 1. Feature Evaluation

Input: clusters C_i ($i = 0..n$), R -tree of π_k T_{π_k} .

Output: $\phi_{LB}(C_i, \pi_k)$, $\phi_{UB}(C_i, \pi_k)$, feature list $L(C_i, \pi_k)$.

Description:

```

1: repeat
2:   for each cluster  $C_i$  do
3:     let  $e$  be the non-leaf entry in  $L(C_i, \pi_k)$  with minimal  $d_{LB}(C_i, e)$ ;
4:     for each cluster  $C_j$  containing  $e$  do
5:       replace  $e$  with its children in  $T_{\pi_k}$ ;
6:       remove entries  $e_r$  if  $d_{LB}(C_j, e_r) \geq \phi_{UB}(C_j, \pi_k)$ ;
7:       update  $\phi_{LB}(C_j, \pi_k)$ ,  $\phi_{UB}(C_j, \pi_k)$ ,  $q(C_j, \pi_k)$ ;
8:       if  $q(C_j, \pi_k) \geq \phi_{UB}(C_j, \pi_k)$  then
9:         remove entries  $e_r$  if  $d_{LB}(C_j, e_r) \geq \phi_{UB}(C_j, \pi_k)$ ;
10:      update  $q(C_j, \pi_k)$ ;
11: until entries in  $L(C_i, \pi_k)$  for all  $i$  are leaf entries
    
```

At the beginning of Algorithm 1, we assume the root of T_{π_k} is a candidate for all clusters, and insert it in all lists. In each iteration from Line 2 to Line 10 in Algorithm 1, the lists are visited in a round-robin fashion. A non-leaf entry with the minimum lower bound $d_{LB}(C_i, e)$ is selected for the current list. Here, a non-leaf entry means the corresponding R -tree node is not a leaf node. We replace it by its children in the R -tree in all lists. A child e_r is inserted into C_j 's list, when its low bound $d_{LB}(C_j, e)$ isn't less than $\phi_{UB}(C_j, \pi_k)$, the minimum upper bound of all entries in the list. After updating $\phi_{LB}(C_j, \pi_k)$, $\phi_{UB}(C_j, \pi_k)$ and $q(C_j, \pi_k)$, we verify the list and filter those entries whose lower bounds $d_{LB}(C_j, e)$ is greater than the updated $\phi_{UB}(C_j, \pi_k)$. This verification could be skipped when $q(C_i, \pi_k)$ is between $\phi_{LB}(C_i, \pi_k)$ and $\phi_{UB}(C_i, \pi_k)$. We repeat these steps until there is not any non-leaf entry in all lists.

3.2 Similarity Search

In this section, we will discuss how to compute the exact distances between pairs of clusters and feature groups based on the generated candidate features for answering the SSIOFqueries. Our goal is the find the cluster most similar to the query C_0 while minimizing the computation complexity. Algorithm 2 presents the overview of the similarity search step.

The input parameter $L(C_i, \pi_k)$ is the candidate list for cluster C_i and feature group π_k . Function *ComputeExact*(C_i, π_k) in Line 1 and 5 computes the exact distance between C_i and feature group π_k . C_{min} is the cluster with minimal $Sim_{LB}(C_i, C_0)$. $Sim_{LB}(C_i, C_0)$ and $Sim_{UB}(C_i, C_0)$ denote the lower and upper bound of similarity between C_i and C_0 , as computed from the input as follows.

Firstly $\phi(C_0, \pi_k)$ are pre-computed for all feature groups. Suppose $f_L = \phi_{LB}(C_i, \pi_k) - \phi(C_0, \pi_k)$ and $f_U = \phi_{UB}(C_i, \pi_k) - \phi(C_0, \pi_k)$, then we have

$$LB_{i,j} = \begin{cases} 0 & f_L \times f_U < 0 \\ \min(|f_L|, |f_U|) & otherwise \end{cases}$$

Algorithm 2. Similarity Search

Input: $L(C_i, \pi_k)$ for $i = 0..n, j = 1..d$, cluster set $\{C_i(i = 0..n)\}$

Output: The cluster C_i with minimal $Sim(C_i, C_0)$ ($i \neq 0$).

Description:

- 1: *ComputeExact*(C_0, π_k) for all j ; remove C_0 from cluster set;
 - 2: $result = \infty$;
 - 3: **while** $Sim_{LB}(C_{min}, C_0) \leq result$ **do**
 - 4: **for** all feature groups π_k **do**
 - 5: *ComputeExact*(C_{min}, π_k);
 - 6: **if** $Sim_{LB}(C_{min}, C_0) > result$ **then**
 - 7: break; //from FOR
 - 8: $result = \min\{result, Sim(C_{min}, C_0)\}$;
 - 9: remove C_{min} from cluster set;
 - 10: return all clusters C_i with $Sim(C_i, C_0) = result$.
-

and

$$UB_{i,j} = \max(|f_L|, |f_U|).$$

Thus,

$$Sim_{LB}(C_i, C_0) = \sum_j LB_{i,j}$$

$$Sim_{UB}(C_i, C_0) = \sum_j UB_{i,j}$$

For example, the results of the feature evaluation step, including the lower bound ($\phi_{LB}(C_i, \pi_k)$) and upper bound $\phi_{UB}(C_i, \pi_k)$ are stored in a 2-dimensional array as shown in Figure 4. The initial bound of similarity between C_i and C_0 are computed as shown on the last column.

	π_1	π_2	π_3	$Sim(C_i, C_0)$
C_0	6	2	9	[0, 0]
C_1	[3, 4]	[11, 12]	[8, 9]	[11, 14]
C_2	[8, 12]	[1, 11]	[9, 14]	[2, 20]
C_3	[4, 7]	[2, 4]	[10, 10]	[1, 5]

Fig. 4. Similarity Evaluation

The clusters are sorted on lower bound of $Sim(C_i, C_0)$ and iteratively computed for precise similarity, until the next lower bound is larger than an already-found result. This sequence can minimize the number of clusters that is precisely computed. Also in Line 6, after each calling of *ComputeExact*, the current lower bound of similarity is refined using the exact distance returned from the function, and is compared with the result, which greatly reduce the number of feature groups need to be computed. For the example in Fig. 4, the cluster

C_3 is first chosen since lower bound of $Sim(C_3, C_0)$ is the minimal in all clusters. Suppose its precise similarity is 3. The next cluster is C_2 . After calling $ComputeExact(C_2, \pi_1)$, assume the lower bound of $Sim(C_2, C_0)$ is updated to be 4, which is larger than the current result 3. As a result, C_2 is dropped as it can not be the result. Also the lower bound of $Sim(C_1, C_0)$ is larger than the current result 2, and C_1 is eliminated as well and the final result C_3 is returned.

Lemma 2. *Algorithm 2 gives the correct answer to the similarity search query.*

Proof. Consider the case that Algorithm 2 returns C_i as result and the exact answer is C_j where $i \neq j$. This is impossible since after C_i is precisely computed, the lower bound of $Sim(C_j, C_0)$ must be smaller than $Sim(C_i, C_0)$, in consequence, C_j is chosen to be precisely computed and C_j should be returned instead of C_i . □

Also, it is easy to see that Algorithm 2 minimize the number of chosen clusters. Suppose that the cluster returned is C_r with result r , and there exists an algorithm A which minimizes the number of chosen clusters. In algorithm A , a cluster C_i such that lower bound of $Sim(C_i, C_0)$ is larger than r must not be visited while all other clusters must be considered for precise computation. This is exactly the case of Algorithm 2. For a cluster C_i that $Sim_{LB}(C_i, C_0) > r$, $Sim_{LB}(C_i, C_0) > Sim_{LB}(C_r, C_0)$. Thus in Algorithm 2, C_r is chosen before C_i . After processing C_r , $result$ is updated to r and C_i are dropped.

Edge Pruning

$ComputeExact(C_i, \pi_k)$ is used to compute the exact distances between a cluster C_i and a feature group π_k . It need to calculate all the distance between the points in C_i and the candidate features in every feature groups. The brute-force way is to compute the distance between a point and every edge in some feature and choose the minimum one as the distance of the point to the feature. We proposed some techniques that can avoid useless computations and save much more time than the brute-force way.

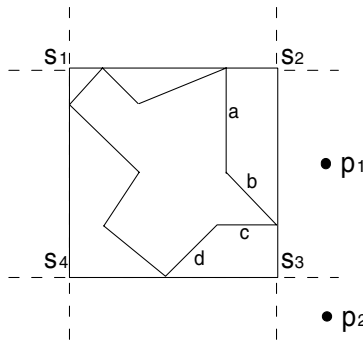


Fig. 5. Edge Pruning

The optimization comes from reducing feature edges need to be computed. As shown in Figure 5, the rectangle is the minimum bounding rectangle of a certain feature. By extending the four edges of the MBR, we partition the whole space into 8 areas except the MBR itself. s_1, s_2, s_3 and s_4 are vertices of the MBR and a, b, c and d are four edges on the feature. p_1 and p_2 are points belonging to some cluster.

Take p_1 as an example. It need calculating all the distance between p_1 and all edges of the feature in the brute-force way. In fact, we can found that the minimum distance from p_1 to the feature must be the minimum distance of p_1 to one of the four edges a, b, c and d . In case of p_2 , the minimum distance from p_2 to the feature must be the minimum distance of p_2 to one of the two edges c and d .

To formalize, if the project of a point p_k to the closest edge $s_i s_j$ of the MBR falls in the edge, then we only need to compute such kind of edges that $s_i s_j$ can be project on. If not, suppose the nearest vertex of MBR to p is s_i , only the edges that s_i can be project on are computed. To further reduce the time complexity, edge projections of a feature are computed at most once and then stored in memory for all other points. Also, when the MBR of a cluster is wholly contained in one of the 8 areas, it is not necessary to check the position of each point any more.

Extend to k -Clusters

The above algorithm is extended to return the k clusters which are most similar to the given cluster C_0 . In Line 8 of Algorithm 2, variant *result* should be set to the k -th lowest similarity, and k most similar clusters are returned in Line 13.

4 Discussion

As mentioned in the above section, for a node on the R^* -tree, we visit it at most once. In each step, the node to be visited is chosen considering only one cluster while ignoring the preference of other clusters. This searching strategy is based on an assumption that the number of clusters is relatively small, since the strategy sacrifices local optimization for each cluster to achieve a better global I/O cost. Since reading disk is much more costly than in-memory computation, our algorithm works well when the number of clusters is not too large.

For the case that the number of clusters is so large that the sacrifice of local computation is unbearable, we can use following divide-and-conquer strategy which is similar to the Nested Loops Join. We first partition the clusters into several parts by grouping near clusters. Then we use our proposed algorithm on each part of clusters. In this case, if there are n groups of clusters, each node of an R^* -tree is visited for at most n times.

5 Experiments

We implemented our proposed *ES* algorithm and evaluate its performance on synthetic data. We use the algorithm *CPM* (Compute Proximity Matching) as

a benchmark based on [9]. The algorithm *CPM* solves a problem that is similar to our problem assuming the number of feature polygons is relatively small and there is no spatial index built on the features. It reads the relevant clusters into buffer first, then read features batch by batch into buffer and determine their groups. For each cluster C_i and feature group π_j , it computes the approximate distance between C_i and each feature in π_j for filtering out features that are too far from C_i . Maintain a list of candidate features for computing $\phi(C_i, \pi_j)$. Then it computes the approximate similarity for each cluster and filter out clusters that are not the solution. Finally it calculates the exact similarities to the remaining clusters and their associate features, and return the query result.

Suppose the number of cluster is n and the number of features is m . Feature number is the same in each of the g features groups. The number of points in each cluster is nc and nf gives the number of edges in each feature polygon. In the experiments, average nc is 100 and average nf is 15.

To generate data, we firstly generate $m + n$ rectangles that are uniformly distributed in the 2-dimensional space. The size of rectangles are randomly chosen within a limited range. Number of features in each group is determined such that the summary is m . Rectangles corresponding to the clusters or a features group do not intersect with each other. In each of n rectangles, nc points are uniformly generated, based on which the *MBRs* are computed. This gives the nc clusters. In each of the remaining rectangles, nf points are randomly generated. To generate a simple polygon which is linked by the nf points. We will apply a Graham's scan-like algorithm [3].

We use *R**-trees, a variant of *R*-tree, to index the feature groups. Two algorithms *CPM* and *ES* are implemented using C++, Experiments are run on a Linux machine with 1.8G P4 CPU and 512M memory. For each dataset, we process extensive queries and get the average result.

5.1 Scalability Comparison

We first compare the algorithms with different number of clusters, ranging from 200 to 1000. 100000 features are clustered in 10 groups. The experiment results are shown in Fig. 6.

The Fig. 6 (a) compares the I/O cost, which is the summary of the number of pages that corresponding to *R**-tree index and features. *CPM* does not use index, but reads a large amount of features; *ES* reads a small number of index pages in the first step and a few features in the second step. It is clear that the I/O cost of *ES* is much smaller than *CPM*.

User time for precessing a query is compared in Fig. 6 (b). For 1000 clusters, *ES* responds in about 42 seconds while *CPM* needs nearly 5 minutes to get the result.

We also study the performance of both algorithms with different number of features. Feature number varies from 5000 to 100000. There are 10 feature groups and the number of clusters is set to 200. Results are shown in Fig. 7. Similar with the previous experiments, the first sub-figure shows the I/O cost while the second compares the precessing time.

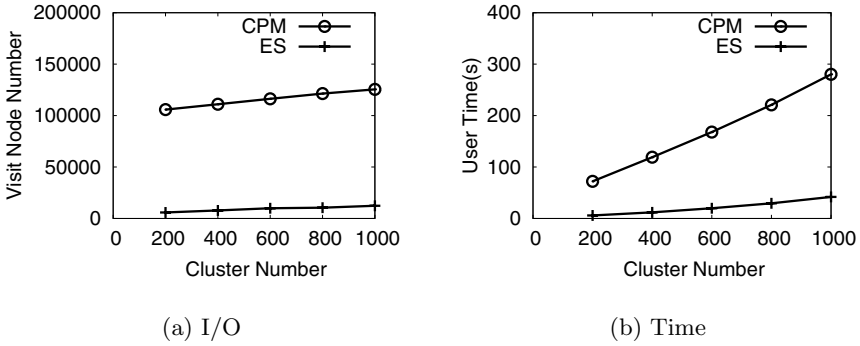


Fig. 6. Compare Cluster Number

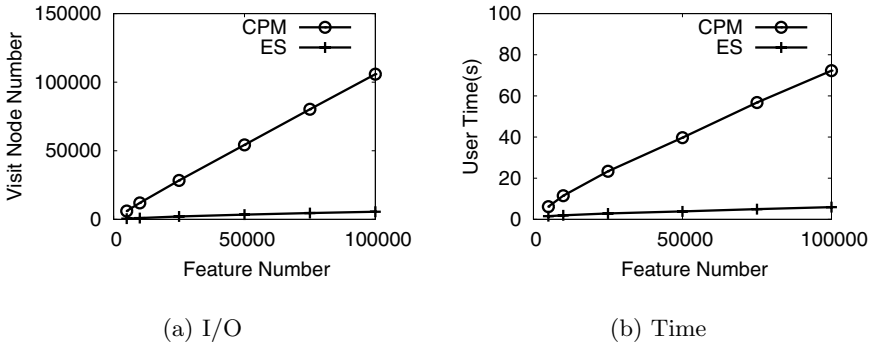


Fig. 7. Compare Feature Number

With 100000 features, our algorithm processes a query in 6 seconds and less than 6000 disk pages read in memory, compared with large I/O cost and more than 1 minute processing time of *CPM*.

5.2 Dimensionality Comparison

We evaluate our algorithm with different dimensionality of feature space. The number of feature group varies from 2 to 20. 100000 features are categorized to the feature groups and number of clusters is 200. Fig. 8 shows the I/O cost and user time of the two algorithms, which demonstrates the large performance difference between the two algorithms.

6 Conclusions

In this paper, a similarity search problem which is based on an implicit feature space is investigated. By making use of the spatial indexes like *R*-trees built on the feature categories, we present an effective algorithm for the queries, which consists two steps: feature evaluation and similarity search. Experiments show the efficiency of the algorithm on all cases.

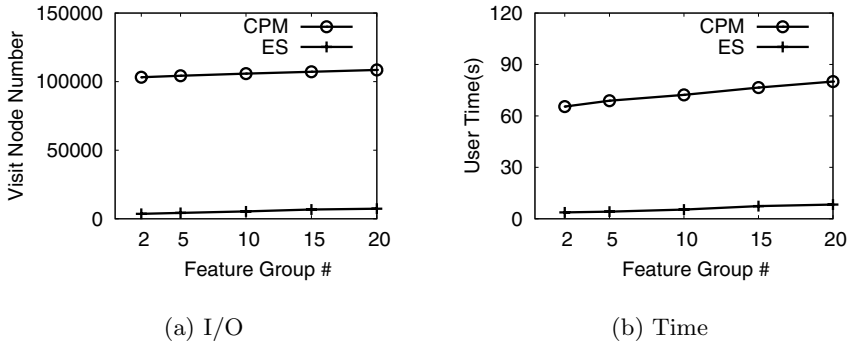


Fig. 8. Compare Feature Group Number

For the future work, we will investigate the problem of similarity join, which joins a set of clusters to itself, with respect of d different categories of features.

Acknowledgment. The research described in this paper was partially supported by ARC Discovery Grant (DP0346004).

References

1. Yuhan Cai and Raymond Ng. Indexing spatio-temporal trajectories with chebyshev polynomials. In *Proceedings of the 2004 ACM SIGMOD*, pages 599–610, 2004.
2. Antonio Corral, Yannis Manolopoulos, Yannis Theodoridis, and Michael Vassilakopoulos. Closest pair queries in spatial databases. In *Proceedings of the 2000 ACM SIGMOD*, pages 189–200, 2000.
3. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry - Algorithm and Applications*. Springer-Verlag, Berlin, 1997.
4. Christos Faloutsos, Ron Barber, Myron Flickner, Jim Hafner, Wayne Niblack, Dragutin Petkovic, and William Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231–262, 1994.
5. Gísli R. Hjaltason and Hanan Samet. Distance browsing in spatial databases. *ACM Trans. Database Syst.*, 24(2):265–318, 1999.
6. H. V. Jagadish. A retrieval technique for similar shapes. In *Proceedings of the 1991 ACM SIGMOD*, pages 208–217, 1991.
7. Tamer Kahveci, Ambuj K. Singh, and Aliekber Gürel. Similarity searching for multi-attribute sequences. In *Proceedings of the 14th SSDM*, Washington, DC, USA, 2002.
8. Hans-Peter Kriegel, Stefan Brecheisen, Peer Kröger, Martin Pfeifle, and Matthias Schubert. Using sets of feature vectors for similarity search on voxelized CAD objects. In *Proceedings of the 2003 ACM SIGMOD*, pages 587–598, 2003.
9. Xuemin Lin, Xiaomei Zhou, and Chengfei Liu. Efficient computation of a proximity matching in spatial databases. *Data Knowledge Engineering*, 33(1):85–102, 2000.
10. Apostol Natsev, Rajeev Rastogi, and Kyuseok Shim. WALRUS: a similarity retrieval algorithm for image databases. In *Proceedings of the 1999 ACM SIGMOD*, pages 395–406, 1999.
11. Nick Roussopoulos, Stephen Kelley, and Frédéric Vincent. Nearest neighbor queries. In *Proceedings of the 1995 ACM SIGMOD*, pages 71–79, 1995.

An Improved FloatBoost Algorithm for Naïve Bayes Text Classification

Xiaoming Liu, Jianwei Yin, Jinxiang Dong, and Memon Abdul Ghafoor

Department of Computer Science and Technology, Zhejiang University, China
{liuxiaoming, zjuyjw,djx}@zju.edu.cn, ghafoorgem@yahoo.com

Abstract. Boosting is a method for supervised learning, which has successfully been applied to many different domains and has proven one of the best performers in text classification exercises so far. FloatBoost learning uses a backtrack mechanism after each iteration of AdaBoost learning to minimize the error rate directly, rather than minimizing an exponential function of the margin as in the traditional AdaBoost algorithm. This paper presents an improved FloatBoost boosting algorithm for boosting Naïve Bayes text classification, called DifBoost, which combines Divide and Conquer Principal with the FloatBoost algorithm. Integrating FloatBoost with the Divide and Conquer principal, DifBoost divides the input space into a few sub-spaces during training process and the final classifier is formed with the weighted combination of basic classifiers, where basic classifiers are affected by different sub-spaces differently. Extensive experiments using benchmarks are conducted and the encouraging results show the effectiveness of our proposed algorithm.

1 Introduction

Text classification is the activity of automatically building, by means of machine learning techniques, automatic text classifiers, i.e. programs capable of labeling natural language texts with thematic categories from a predefined class set. A wealth of different methods have been applied to it, including probabilistic classifiers, decision trees, decision rules, regression methods, batch and incremental linear methods, neural networks, example-based methods, and support vector machines (See [2] for a review). In recent years, the method of classifier committees has also gained popularity in the text classification community.

The boosting method [1] occupies a special place in the classifier committees literature. Since the boosting technique was developed [1], it has been considered to be one of the best approaches to improving classifiers in many previous studies. In particular, boosting contributes to significantly improve the decision tree learning algorithm [3,4]. FloatBoost [6] is an improved AdaBoost method for classification, which incorporates into AdaBoost the idea of Float Search, originally specified in [5] for feature selection. FloatBoost achieves a stronger classification consistency of fewer weak classifiers than AdaBoost and has shown its performance in face detection field [6].

When boosting is used to handle scenarios in complex environment with outliers, its limitations have been pointed out by many researchers [4,7], some discussion and approaches have been proposed to address these limitations [8,9]. In [8], S-AdaBoost algorithm which applying the Divide and Conquer Principle to the AdaBoost algorithm was proposed to enhance AdaBoost’s capability of handling outliers in face detection field. In this paper, we focus on boosting Naïve Bayes classifier, which is a simple yet surprisingly accurate technique and has been used in many different classification problems. In particular, for text classification, Naïve Bayes classifier is known to be remarkably successful despite the fact that text data generally has a huge number of attributes (features). By integrating Divide and Conquer Principle with FloatBoost for boosting Naïve Bayes text classifier, we propose an improvement FloatBoost algorithm, called DifBoost.

The rest of the paper is organized as follows. In section 2, preliminary backgrounds are introduced. In Section 3, we describe in detail our proposed DifBoost algorithm. The results of its experimentation and comparisons between DifBoost and other methods are described in Section 4. In section 5, we conclude and predict future work.

2 Preliminaries

2.1 Naïve Bayes Learning Framework for Text Classification

Bayes method assumes a particular probabilistic generation model for text classification. That is, every document is assumed to be generated according to a probability distribution defined by a set of parameters, denoted by θ . The probability distribution consists of a mixture of components $c_j \in C = \{c_1, \dots, c_{|C|}\}$ and each component is parameterized by a disjoint subset of θ . To classify a given document, Bayes learning method estimates the posterior probability of a class via Bayes rules, that is,

$$\Pr(c_j | d_i, \theta) = \frac{\Pr(c_j | \theta) \Pr(d_i | c_j, \theta)}{\Pr(d_i | \theta)}$$

The class identity of document d_i is the class with the most

posterior probability: $\text{argmax}_{c_j \in C} \Pr(c_j | d_i, \theta)$. Usually, a document d_i is represented by a bag of words $(w_{i1}, w_{i2}, \dots, w_{i|d_i|})$. Moreover, Naïve Bayes classifier assumes a simplification that words independence and words position independence, which

results in the following classification function $f_{\theta_{NB}}^{\wedge}(d_i) = \text{argmax}_{c_j \in C} \Pr(c_j | d_i) \prod_{k=1}^{|d_i|} \Pr(w_{ik} | c_j)$.

To generate this classification function, Naïve Bayes learning estimates the parameters of the generative model using a set of labeled training data $D = \{d_1, \dots, d_{|D|}\}$.

The estimate of θ is written as $\hat{\theta}$. Naïve Bayes uses the maximum a posteriori (MAP) estimate, thus finding $\text{argmax}_{\theta} \Pr(\theta | D)$. Which is the value of θ that is most probable

Table 1. The FloatBoost Algorithm with Naïve Bayes

Input: (1) Training documents $D^1 = \{ \langle d_i, c_j \rangle | d_i \in D, c_j \in C \}$. (2) Maximum number M_{\max} of weak classifiers. (3) Acceptance threshold ε^* .

Output: A classifier function $f_{\hat{\theta}_{NB}}$ where $\hat{\theta}_{NB} = \{ \hat{\theta}_{wlc}, \hat{\theta}_c \}$

$$l_{NB}(D^M) = \arg \max_{\hat{\theta}_{NB}} \Pr(D^M | \hat{\theta}_{NB}) \Pr(\hat{\theta}_{NB}) \text{ /* MAP estimate */}$$

$$f_{\hat{\theta}_{NB}}(d_i) = \arg \max_{c_j \in C} \Pr(c_j | d_i, \theta) = \arg \max_{c_j \in C} \Pr(c_j) \prod_{k=1}^{M_j} \Pr(w_k | c_j),$$

$$W^{(M)} = (w_1^{(M)}, \dots, w_{|D|}^{(M)}). \text{ /* weight distribution */}$$

1. Initialize : (1) $w_{di}^{(1)} = 1/|D|$ for any $d_i \in D$;
- (2) $\varepsilon_m^{\min} = \max\text{-value}$ (for $m=1, \dots, M_{\max}$), $M=0, H_0 = \{ \}$
2. Forward Inclusion:

- (1) $M=M+1$, estimate a class model with respect to the weighted training documents, $\hat{\theta}_{NB} = l_{NB}(D^M)$;

- (2) Build a base classifier $h_M = f_{\hat{\theta}_{NB}}^{(M)}(d_i)$ with estimated model $\hat{\theta}_{NB}$;

- (3) Calculate the weighted training error $\varepsilon(h_M)$ of $\hat{\theta}_{NB}$,

$$\varepsilon_M = \varepsilon(h_M) = \sum_{d_i \in D^M} [w_{di}^{(M-1)} I(f_{\hat{\theta}_{NB}}^{(M)}(d_i) \neq f_{\theta}(d_i))];$$

- (4) Calculate confidence α_M of $\hat{\theta}_{NB}$, $\alpha_M = \frac{1}{2} \ln(e^{-\varepsilon_M} / \varepsilon_M)$, and update weights

$$w_{d_i}^{(M)} = \frac{w_{d_i}^{(M-1)}}{Z_M} \times \begin{cases} \exp(-\alpha_M) & \text{if } f_{\hat{\theta}_{NB}}^{(M)}(d_i) = f_{\theta}(d_i) \\ \exp(\alpha_M) & \text{if } f_{\hat{\theta}_{NB}}^{(M)}(d_i) \neq f_{\theta}(d_i) \end{cases}, \text{ where } Z_M \text{ is a normalization}$$

factor making $w^{(M)}$ a probabilistic distribution: $\sum_{d_i \in D^M} w_{d_i}^{(M)} = 1$;

- (5) $H_M = H_M \cup \{h_M\}$, if $\varepsilon_M^{\min} > \varepsilon(H_M)$ then $\varepsilon_M^{\min} = \varepsilon(H_M)$;

3. Conditional Exclusion:

- (1) $h' = \arg \min_{h \in H_M} \varepsilon_{M-1}(H_M - h)$;

- (2) If $\varepsilon(H_M - h') < \varepsilon_{M-1}^{\min}$, then

- (2.1) $H_{M-1} = H_M - h'$, $\varepsilon_{M-1}^{\min} = \varepsilon(H_M - h')$, $M=M-1$, goto 3.(1);

- (3) else

- (3.1) if $M=M_{\max}$ or $\varepsilon_M < \varepsilon^*$, then goto 4;

- (3.2) goto 2.(1);

4. Output the final classifier:

$$f_{\hat{\theta}_{NB}}(d) = \arg \max_{c_j \in C} \sum_{m=1}^M \left[\frac{\alpha_m}{\sum_{n=1}^M \alpha_n} I(f_{\hat{\theta}_{NB}}^{(m)}(d) = f_{\theta}(d)) \right]$$

given the evidence of the training data set and a prior. The estimated probability of a word w_t given a class c_j is the equation:

$$\hat{\theta}_{w_t|c_j} = \Pr(w_t | c_j) = \frac{1 + \sum_{i=1}^{|D|} N(w_t, d_i) \Pr(y_i = c_j | d_i)}{|C| + \sum_{j=1}^{|C|} \sum_{i=1}^{|D|} N(w_t, d_i) \Pr(y_i = c_j | d_i)}$$

Similarly, the class prior probabilities $\hat{\theta}_{c_j}$ are estimated as:

$$\hat{\theta}_{c_j} = \Pr(c_j | \theta) = \frac{1 + \sum_{i=1}^{|D|} \Pr(y_i = c_j | d_i)}{|C| + |D|}$$

2.2 The FloatBoost Algorithm with Naïve Bayes

This section presents the FloatBoost algorithm with Naïve Bayes learning using notation introduced in the previous section and Naïve Bayes learning framework. Different from AdaBoost, FloatBoost backtracks after a newest weak classifier h_M is added and deletes unfavorable weak classifiers h_m from the ensemble, following the idea of Float Search [5] for feature selection.

The FloatBoost procedure is shown in Table 1. Let $H_M = \{h_1, h_2, \dots, h_M\}$ be the so-far-best set of M weak classifiers, $\epsilon(H_M)$ be the error rate achieved by weighted sum of weak classifiers $H_M = \sum_m w^m h_m$, ϵ_m^{\min} be the minimum error rate achieved so far with an ensemble of m weak classifier. In step 2 (forward inclusion), given already selected, the best weak classifier is added one at a time. In step 3 (conditional exclusion), FloatBoost removes the least significant weak classifier from H_M , subject to the condition that the removal leads to a lower error rate ϵ_{M-1}^{\min} . These are repeated until no more removals can be done. The procedure terminates when the error rate is acceptable or the maximum number M_{\max} is reached. Incorporated with the conditional exclusion, FloatBoost usually needs fewer weak classifiers than AdaBoost to achieve the same error rate ϵ .

3 Robust Boosting of Naïve Bayes

3.1 Basic Idea

As mentioned before, to make a classifier capable of handling complex environment with outliers, we should find ways to decrease outliers' effect on the classifier. Our strategy for robust classification is to separate outliers from other patterns. We apply

Divide and Separate Principle [8] through dividing the input pattern space X into a few subspaces and conquering the subspaces by dealing them differently during training weak classifiers. As in [8], input space is divided into 4 subspaces relative to a classifier $f(x)$: $X=X_{no}+X_{sp}+X_{ns}+X_{hd}$, where X_{no} are normal patterns those can be easily classified by $f(x)$, X_{sp} are special patterns those can be classified correctly by $f(x)$ with bearable adjustment, X_{ns} are noise patterns and X_{hd} are patterns hard to be classified by $f(x)$.

A typical input pattern space is shown in Figure 1. The first two subspaces are further referred to as Ordinary Pattern Space and the last two are called Outliers: $X_{od}=X_{no}+X_{sp}$, $X_{ol}=X_{ns}+X_{hd}$. It is relative easier for an algorithm like weak classifiers Naïve Bayes in FloatBoost to classify X_{od} well compared to classify the whole input pattern space X . After the division, weak classifiers can concentrate more on X_{sp} in X_{od} , instead of X_{ol} , which can often improve the generalization of the algorithm.

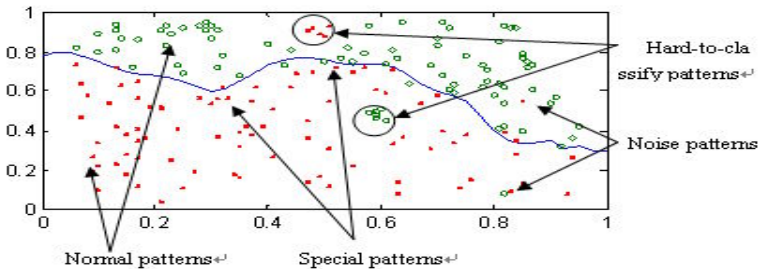


Fig. 1. Input Pattern Space

3.2 Incorporating Divide and Conquer Principle into FloatBoost: DifBoost

To incorporate Divide and Conquer Principle into FloatBoost, a challenging problem is how to isolate outliers X_{ol} from ordinary patterns X_{od} . Given a training data set with some trained weak classifiers, we can see a prominent difference between X_{ol} and X_{od} is that the misclassification count on X_{ol} with weak classifiers is much larger than misclassification count on X_{od} . So a threshold can be used to separate X_{ol} from X_{od} . To improve the accuracy of outlier isolation, isolation is not performed during the initial training stage. In our experiment, $1/2M_{max}$ is often used as a turning point, which means that we do not try to isolate outliers until we have get $1/2M_{max}$ weak classifiers. Furthermore, for the boundary between X_{hd} and X_{od} is often not obvious in practice, we deal X_{ns} and X_{hd} differently. X_{ns} patterns once identified, they will be removed from training set, while X_{hd} patterns will still be used during training. One common thing between X_{ns} and X_{hd} as we can see is that their misclassification rates tend to be high. Meanwhile, a major difference between them is that, a noise pattern will often be misclassified to a specific wrong class, on the other hand, a hard-to-classify pattern will tend be misclassified to different wrong classes.

With the proposed isolation method and treatment of different outlier patterns, table 2 shows modification to the FloatBoost algorithm which is integrated with Divide and Conquer Principle. As shown in clause 2.(3), when we calculate the misclassification

Table 2. Modification to the FloatBoost Algorithm with Naïve Bayes: DifBoost

Input: Three same inputs as in Table 1, in addition, the outlier threshold ϵ_{ol} .

Output: A classifier function $f_{\hat{\theta}_{NB}}$, where $\hat{\theta}_{NB} = \{\hat{\theta}_{wlc}, \hat{\theta}_c\}$

1. Initialize: (1) $w_{d_i}^{(1)}=1/|D|$, $EC_{d_i}=\{\}$ for any $d_i \in D$; /* EC_{d_i} is classifier set that misclassification d_i */

(2) $\epsilon_m^{\min} = \max\text{-value}$ (for $m=1, \dots, M_{\max}$), $M=0$, $H_0 = \{\}$, $X_{ns} = \{\}$, $X_{hd} = \{\}$, $X_{od} = D$;

2. Forward Inclusion:

(1) $M=M+1$, estimate a class model $\hat{\theta}_{NB}^{(M)}$ with respect to the weighted training documents;

$$h_M = f_{\hat{\theta}_{NB}^{(M)}}(d_i)$$

(2) Build a base classifier $f_{\hat{\theta}_{NB}^{(M)}}$ with estimated model $\hat{\theta}_{NB}^{(M)}$;

(3) Calculate the weighted training error $\epsilon(h_M)$,

$$\epsilon(h_M) = \sum_{d_i \in X_{od} - X_{hd}} [w_{d_i}^{(M-1)} I(f_{\hat{\theta}_{NB}^{(M)}}(d_i) \neq f_{\theta}(d_i))] + \frac{1}{2} \sum_{d_i \in X_{hd}} \left[w_{d_i}^{(M-1)} I(f_{\hat{\theta}_{NB}^{(M)}}(d_i) \neq f_{\theta}(d_i)) \right],$$

(3.1) if $M > 1/2 M_{\max}$,

(a) For $\forall d_i \in X_{od}$, if $f_{\hat{\theta}_{NB}^{(M)}}(d_i) \neq f_{\theta}(d_i)$ $EC_{d_i} = EC_{d_i} \cup \{h_M\}$

(b) for $\forall d_i \in X_{od}$, if $EC_{d_i}/M > \epsilon_{ol}$

If $|EC_{d_i}| > 2$, $X_{hd} = X_{hd} \cup \{d_i\}$

Else $X_{ns} = X_{ns} \cup \{d_i\}$, $X_{nd} = X_{nd} - \{d_i\}$

(4) Calculate confidence α_M of $\hat{\theta}_{NB}^{(M)}$, $\alpha_M = \frac{1}{2} \ln(1 - \epsilon_M / \epsilon_M)$, and update weights $w_{d_i}^{(M)}$;

(5) $H_M = H_M \cup \{h_M\}$, if $\epsilon_M^{\min} > \epsilon(H_M)$ then $\epsilon_M^{\min} = \epsilon(H_M)$.

3. Conditional Exclusion:

(1) $h' = \arg \min_{h' \in H_M} \epsilon_{M-1}(H_M - h)$;

(2) If $\epsilon(H_M - h') < \epsilon_{M-1}^{\min}$, then

(2.1) $H_{M-1} = H_M - h'$, $\epsilon_{M-1}^{\min} = \epsilon(H_M - h')$, $M = M - 1$

(a) for $\forall d_i \in X_{hd}$ and $h' \in EC_{d_i}$, if $EC_{d_i}/M < \epsilon_{ol}$, $X_{hd} = X_{hd} - \{d_i\}$;

(b) for $\forall d_i \in X_{ns}$ and $h' \in EC_{d_i}$, if $EC_{d_i}/M < \epsilon_{ol}$, $X_{ns} = X_{ns} - \{d_i\}$,

$X_{nd} = X_{nd} \cup \{d_i\}$;

(2.2) goto 3.(1)

(3) else

(3.1) if $M = M_{\max}$ or $\epsilon_M < \epsilon^*$, then goto 4

(3.2) goto 2.(1)

4. Output the final classifier:

$$f_{\hat{\theta}_{NB}}(d) = \arg \max_{c_j \in C} \sum_{m=1}^M \left[\frac{\alpha_m}{\sum_{s=1}^M \alpha_s} I(f_{\hat{\theta}_{NB}^{(m)}}(d) = f_{\theta}(d)) \right]$$

rates of a weak classifier, patterns in X_{ns} are not taken into account and patterns in X_{hd} are weighted half to patterns in X_{nd} . Whether to put a pattern into X_{ns} or X_{hd} is considered in 2.(3.1) during forward inclusion. Correspondingly, 3.(2) considers whether patterns in X_{ns} and X_{nd} should be reconsidered as ordinary patterns.

In our algorithm, an important parameter is ϵ_{ol} , which is used to determine whether a pattern should be regarded as an outlier. The optional value of ϵ_{ol} is associated with the classification task itself and the nature of patterns in X . Experiments were conducted to determine the optimal value for the threshold ϵ_{ol} . From the experiments conducted, DifBoost performed reasonably well when the value of ϵ_{ol} was around 0.85-0.95.

4 Experimental Setup and Results

In order to evaluate our proposed method, we have conducted experiments on two data sets: the Reuters-21578 collection and 20-Newsgroups Data. Reuters-21578 consists of Reuters newswire stories from 1987, and is the most popular data set in the text classification literature. The data set consists 21,578 articles, each one pre-labeled with one or more of 135 topics. We use the modified Apte split (Mod Apte), which assigns 9,603 documents dated before April 8, 1987 to the training set and 3,299 documents dated from April 8, 1987 to the test set. In our experiments, we use ninety topic categories that have at least one relevant (positive) training documents and at least one relevant test document. The second data set 20-Newsgroups consists of 20,000 Usenet articles collected by K. Lang from 20 different newsgroups. For this data set, about 70% documents in each newsgroup are used for training (700 documents per class), while left documents are used for testing (300 documents per class).

We preprocess both data sets by removing the low-frequency words, which are the words appear less than 2 times in a document. Stop-words are removed and term space reduction is applied. After such a reduction, each (training or test) document d_i is represented by a vector $\langle t_{i1}, \dots, t_{ir} \rangle$ of the weights shorter than the original. Feature selection is usually beneficial in that it tends to reduce both overfitting and the computation cost of training the classifier. We use the information gain [9] for term space reduction. If not specially mentioned, the number of features in our experiment is 600. We use macro-average F1 and micro-average F1 as [9] the evaluation measures of the text classifiers.

Table 3 shows a comparison of the performances of 4 different classifiers on our data sets Reuters and 20-Newsgroups respectively. All the parameters for different classifiers are tuned to yield the best performance. For AdaBoost, FloatBoost and DifBoost, we set training around M_{max} to be 400 in both data sets. The parameter ϵ_{ol} in DifBoost is set to be 0.9. The experimental results indicate that FloatBoost Performances.better.than .AdaBoost, while DifBoost gains most prominent performance, which outperforms FloatBoost further.

Table 3. Performances of four different classifiers

	Macro-average F1		Micro-average F1	
	Reuters	20-Newsgroups	Reuters	20-Newsgroups
Naïve Bayes(NB)	0.785	0.804	0.796	0.798
AdaBoost(AB)	0.802	0.817	0.805	0.816
FloatBoost(FB)	0.822	0.832	0.816	0.826
DifBoost(DB)	0.852	0.843	0.837	0.854

Results of different methods on both data sets are shown in Figure 2-5. Figure 2 and 3 show the effectiveness of individual methods on part of Reuters evaluated by Macro-average F1 and Micro-average F1 respectively. Figure 4 and 5 shown effectiveness of each method on 20-Newsgroups. The X-axis of each figure represents the number of training documents. To explore the capacity of handling outliers of DifBoost method, in Reuters we use only 10 largest and 10 smallest categories from the ninety categories. Experimental parameters are set as follows, M_{\max} in three boosting algorithm are set to be 400, ϵ_{ol} is 0.9 in DifBoost and optional parameter ϵ^* in DifBoost is not set, which defaults to be 0.

As shown in Figures 2-5, we have observed that the proposed DifBoost method is successful in boosting Naïve Bayes. AdaBoost could increase the quality of Naïve Bayes classifier with an average increase about 10% in both F1 measures over pure Naïve Bayes algorithm (NB). FloatBoost could increase the quality of Naïve Bayes classifier with an average increase about 16% in both F1 measures and DifBoost outperforms Naïve Bayes about 20% in both F1 measures. Note in some cases, AdaBoost is worse than Naïve Bayes in our experiments, the phenomenon was also observed in previous experiments [10]. On our selected sub Reuters data set, DifBoost performance much better than all the other three methods. The Macro-average F1 measure of DifBoost is about 25% better than Naïve Bayes and 10% better than FloatBoost, and the Micro-average F1 measure of DifBoost is 15% better than FloatBoost on sub Reuters data set. Experimental results indicate that DifBoost performs best with medium-size training set.

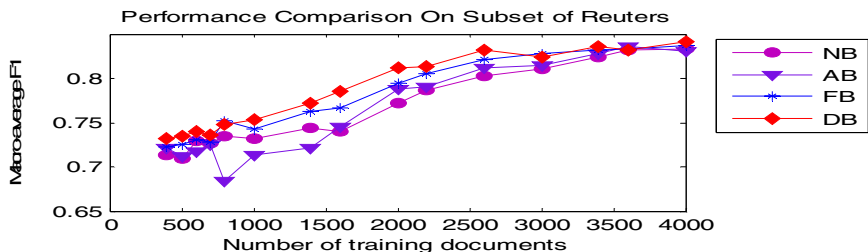


Fig. 2. Macro-average F1 of Naïve Bayes, AdaBoost, FloatBoost and DifBoost with different training size on subset of Reuters

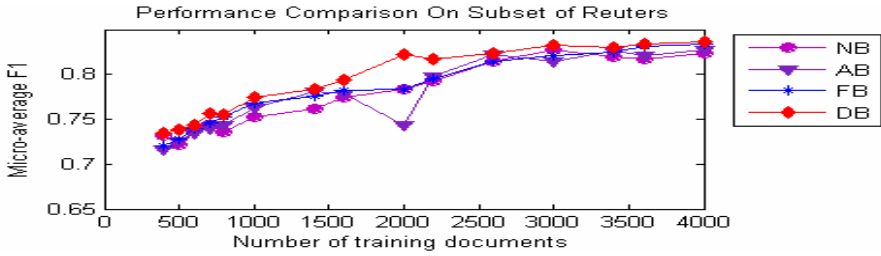


Fig. 3. Micro-average F1 of Naïve Bayes, AdaBoost, FloatBoost and DifBoost with different training size on subset of Reuters

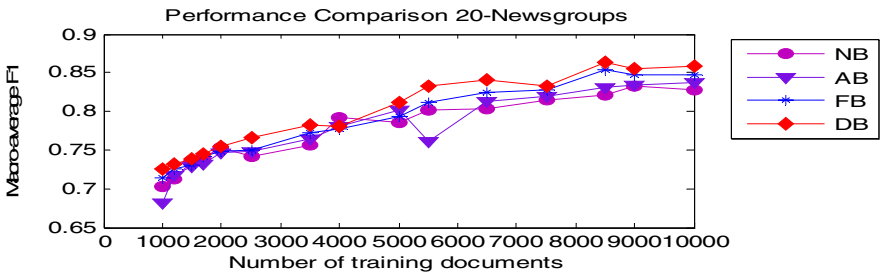


Fig. 4. Macro-average F1 of Naïve Bayes, AdaBoost, FloatBoost and DifBoost with different training size on 20-Newsgroups

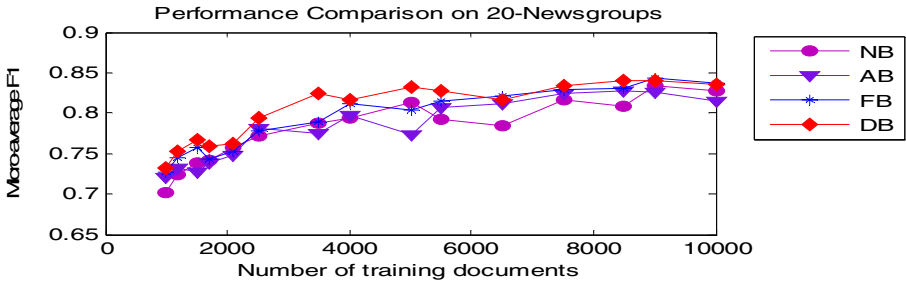


Fig. 5. Micro-average F1 of Naïve Bayes, AdaBoost, FloatBoost and DifBoost with different training size on 20-Newsgroups

5 Conclusions and Future Work

We have described DifBoost, a boosting algorithm derived by FloatBoost with Naïve Bayes by integrating with the Divide and Conquer Policy, and we have reported the results of its experimentation on Reuters-21578 and 20-Newsgroup data sets. The basic idea behind our method is to increase the capability of FloatBoost algorithm to handle outliers in the field of text classification. To this end, we have endowed the FloatBoost

algorithm with the capacity of outlier detecting and handling. Experimental results show the effectiveness of the proposed algorithm. In the future, we plan to combine kNN, support vector machine algorithms with DifBoost algorithm since they are long used and are effective in text classification also.

References:

1. Freund, Y. and Schapire, R. E. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. Proceedings of the 2th European Conference on Computational Learning Theory.
2. Sebastiani, F. 2002. Machine Learning in Automated Text Categorization. ACM Computing Surveys. 34(1):1-47
3. Freund, Y. and Schapier, R.E., 1996. Experiments with a New Boosting Algorithm. International Conference on Machine Learning. 148-156
4. Friedman, J.H., Hastie, T., and Tibshirani, R., 2000. Additive logistic regression: A statistical view of boosting. Annals of Statistics. 28(2):337-374
5. Pudil, P., Novovicova, J. and Kittler, J. 1994. Floating search methods in feature selection. Pattern Recognition Letters, (11):1119-1125
6. Li, S.Z., Zhang, Z.Q. 2004. FloatBoost Learning and Statistical Face Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(9):1112-1123
7. Jiang, W. 2001. Some theoretical aspects of boosting in the presence of noisy data. Proceedings of the Eighteenth International Conference on Machine Learning. 234-241
8. Jimmy, L.J., Loe, K.F. 2003. S-AdaBoost and Pattern Detection in Complex Environment. Proceeding of CVPR, 413-418
9. Yang, Y. and Liu, X. 1999. A re-examination of text categorization methods. In Proceedings of SIGIR-99, pp.42-49
10. Kim, H. and Kim, J. 2004. Combining Active Learning and Boosting for Naïve Bayes Text Classifiers. Proceeding of WAIM 2004, LNCS 3129, pp.519-527

An Approach to RDF(S) Query, Manipulation and Inference on Databases

Jing Lu, Yong Yu, Kewei Tu, Chenxi Lin, and Lei Zhang

APEX Data and Knowledge Management Lab,
Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai, 200240, China
{robertlu, yyu, tkw, linchenxi, zhanglei}@apex.sjtu.edu.cn

Abstract. In order to lay a solid foundation for the emerging semantic web, effective and efficient management of large RDF(S) data is in high demand. In this paper we propose an approach to the storage, query, manipulation and inference of large RDF(S) data on top of relational databases. Specifically, RDF(S) inference is done on the database in advance instead of on the fly, so that the query efficiency is maximized. To reduce the cost of inference, two inference modes, the batch mode and the incremental mode, are provided for different scenarios. In both modes, optimized strategies are applied for efficiency purpose. In order to support efficient query and inference on the database, the storage schema is also specially designed. In addition, a powerful RDF(S) query and manipulation language RQML is provided for easy and uniform data access in a declarative way. Finally, we evaluate and report the performance on both query and inference of our approach. Experiments show that our approach achieves encouraging performance in million-scale real data.

1 Introduction

The semantic web is emerging as the next generation web, where web contents could be understood by machines. RDF(S) is a W3C standard for the formalization of information on web resources, laying one of the foundations of the semantic web. Thus effective and efficient management of RDF(S) data is a must, including storage, query, manipulation and inference.

Since scalability is of great concern in real-world applications, it is necessary to support the management of RDF(S) data in large volumes. Therefore, in this paper we propose an approach to the storage, query, manipulation and inference of RDF(S) on top of the relational database, which is the primary choice to manage large-volume data in practice.

A main feature of our approach is that, to maximize the query efficiency on inferred data, we store all the triples that are deduced from the RDF(S) closure computing, instead of calculating them on the fly. This is extremely imperative if a large data set is presented or the query is quite complex. One consequence of this decision is that we have to do inference on each manipulation, so as to maintain the consistence and completeness of the inferred data. Two modes

of inference, i.e. batch mode and incremental mode, are therefore developed to reduce the maintenance cost according to different manipulation scenarios. In addition, we introduce the Original Semantics Assumption and accordingly circumvent mutual dependency loops contained in the RDF(S) entailment rules while performing inference. The database schemas in our approach are elaborately designed so that not only query but also inference is well supported. We also introduce a language called RQML, which provides powerful and flexible query ability as well as manipulation ability. Finally, we report our performance evaluation, which shows that in a typical scenario the inference time complexity is around $O(n^2)$, and the query time complexity is around $O(n)$.

We have implemented our approach as the base layer of an integrated ontology engineering environment called ORIENT [1], which has been released at the IBM AlphaWorks website¹.

The rest of the paper is organized as follows. Section 2 presents the database schemas we adopted to store RDF(S) data. Section 3 discusses our inference algorithm on databases, including the batch and incremental mode. Section 4 introduces RQML, our RDF(S) query and manipulation language. The performance of our approach on query and inference is evaluated in Section 5. Finally we discuss the related work in Section 6 and conclude the paper in Section 7.

2 Database Schemas for Storage

In order to support efficient RDF(S) query and manipulation processing as well as RDF(S) inference (i.e. RDF(S) closure calculation), we carefully designed a relational database schema to store RDF(S) ontology. We followed the following principles in the design of the storage schema:

- For high-performance query, manipulation and inference, we trade storage space for speed.
- Although the storage schema should be able to handle arbitrary RDF(S) data, it must be optimized for normal data distribution and typical queries in ontology engineering scenarios.

Table.1 lists the database tables in the current storage schema design. The design of the tables are mainly influenced by and closely related to the RDF(S) closure computing algorithm, which is introduced in detail in the next section.

On the other hand, queries can also be efficiently carried out on these tables. Typical queries about class hierarchy, property hierarchy/domain/range and instance type can be answered quickly by directly querying the corresponding table. In addition to these tables for certain RDF schema triples, the `RDFLiteralInteger`, `RDFLiteralFloat`, `RDFLiteralBoolean` and `RDFLiteralString` tables are created to hold common data type literals and leverage native SQL data type comparison and calculation, which is required to support some kinds of queries. We also borrow Jena2's Property Tables design [2] in

¹ <http://www.alphaworks.ibm.com/tech/semanticstk>.

Table 1. Database tables

Table Name	Table Description
RDFSubPropSubProp	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, $\text{rdfs:subPropertyOf}$).
RDFSubPropDomain	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, rdfs:domain).
RDFSubPropRange	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, rdfs:range).
RDFSubPropSubClass	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, rdfs:subClassOf).
RDFSubPropType	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, rdf:type).
RDFSubProp	This table holds all triples ($?x$, $\text{rdfs:subPropertyOf}$, $?y$).
RDFDomain	This table holds all triples ($?x$, rdfs:domain , $?y$).
RDFRange	This table holds all triples ($?x$, rdfs:range , $?y$).
RDFSubClass	This table holds all triples ($?x$, rdfs:subClassOf , $?y$).
RDFType	This table holds all triples ($?x$, rdf:type , $?y$).
RDFStatement	This table holds all triples ($?x$, $?y$, $?z$).
RDFResource	This table holds the URI or blank node ID for all RDF resources.
RDFLiteralInteger	This table holds all literals with type xsd:integer .
RDFLiteralFloat	This table holds all literals with type xsd:float .
RDFLiteralBoolean	This table holds all literals with type xsd:boolean .
RDFLiteralString	This table holds all literals with type xsd:string .
RDFTypedLiteral	This table holds all typed literals that are not contained in the above literal tables.
RDFPlainLiteral	This table holds all plain literals.
RDFUserProp	This table holds the names of user specified properties.
RDFProp*	These tables hold user specified property groups.

the `RDFUserProp` and `RDFProp*` tables to speed up the queries on certain user specified properties.

The statement tables (i.e. the first eleven tables), which hold RDF(S) triples, contain a flag column which can take one of the following values: `EXPLICIT`, `DERIVED` and `SUSPENDED`. The value `EXPLICIT` and `DERIVED` indicate whether the statement is explicitly declared or is derived by inference. The value `SUSPENDED` is a temporary value which will be used in the process of inference.

Note that the RDF data are redundantly stored in the statement tables. For example, a statement like ($?x$, rdf:type , $?y$) exists both in the `RDFType` table and `RDFStatement` table. In addition, as mentioned above, we store derived statements together with explicitly declared statements in the tables. This can also be seen as a kind of redundancy. However, this redundancy greatly facilitates and speeds up query processing.

- As derived statements are stored together with the explicitly declared statements, answering queries involving inference is almost as quick as answering queries without inference.
- Since the data contained in each table are complete by itself, the minimum number of tables are need to support a query. In this way, table join operations are greatly reduced. For example, the `RDFSubProp` table itself is able to support the property hierarchy queries.
- For most simple queries, only a single statement table is involved. As a result, a single SQL sentence can answer the query and hence we could simply utilize some SQL functions (e.g. ordering) provided by DBMS which are more efficient than those implemented by ourselves outside DBMS.

3 Inference on Databases

Reasoning on existing knowledge to discover implicit information is an important process on the Semantic Web. Common queries, such as “what are the (direct and indirect) sub-classes and instances of an exiting class” and “which instances have a certain relationship with a given instance”, may all involve inference.

In real-world applications, most users wish to view a knowledge base at the semantic level, that is they want to query the derived data together with the explicitly declared ones. Currently in most RDF(S) management systems, the inference is not performed until it is imperative to answer a query. However, answering queries in this way might be time-consuming, especially when the data amount is large or the query is complex, which is the case in practice. Therefore, we choose to achieve better query performance at the cost of larger storage size. In other words, we choose to compute the complete RDF(S) closure and store all the derived RDF(S) statements in the database together with the explicit RDF(S) statements. When the knowledge base is modified, inference would be performed to maintain the consistency and completeness of the inferred data.

Our inference supports a core subset of the RDF(S) entailment rules including `rdf1`, `rdfs2` – `rdfs11`, `rdfs13` (as defined in the RDF Semantics document [3]). We call these twelve rules *the rule set* and the involved five RDF(S) properties (`rdfs:domain`, `rdfs:range`, `rdfs:type`, `rdfs:subPropertyOf` and `rdfs:subClassOf`) *the reserved property vocabularies*. This rule set is selected based on the entailment rules’ importance and usage in common RDF ontology engineering scenarios.

In order to reduce the inference cost for different scenarios, two inference modes are provided, i.e. the batch mode and the incremental mode. In the batch mode, RDF(S) closure is not computed until the system is told to do so. This mode is suitable for batch update to the RDF data, since batch update may lead to a large amount of closure computation that would cost a long time. We design an optimized algorithm for this mode to compute the RDF(S) closure. In section 3.1, we will give the detailed description of this algorithm.

In the incremental mode, inference is performed whenever there are updates to the RDF data. The algorithm for the incremental mode is more like a forward-chaining closure computing method. However, since the forward-chaining closure computing method does not support retractions, we design an special algorithm for this purpose. We will give the detailed description in section 3.2.

3.1 Inference in the Batch Mode

A brute force method of calculating the closure is to repetitively grow the RDF knowledge base according to the rule set until a fix point is reached. This straightforward method, however, is very inefficient, especially when it is performed on a relational database. By analyzing the calculation process, we found that the inefficiency of the process mainly stems from the following problems:

P1: The two transitive predicates, `rdfs:subPropertyOf` and `rdfs:subClassOf`, give rise to repetitive calculation.

- P2:** The rules in the rule set, such as `rdfs7`, and the `rdfs:subPropertyOf` predicate create interdependent derivation relationships among the predicates. The iteration caused by such interdependency relationships is the main source of the inefficiency.
- P3:** On a relational database, the insertion of triples one by one is much less efficient than using batch SQL commands.

To better explicate first two problems, we present here a simplified graph of the predicates' derivation relationships caused by the rule set (Fig.1). Edge labels are names of the rules that cause the derivation relation. Two set of derivation relations are omitted from the graph:

- R1:** Rules `rdf1`, `rdfs2`, `rdfs3`, `rdfs4a` and `rdfs4b` may derive `rdf:type` from any predicate.
- R2:** Rule `rdfs7` may derive any predicate from any predicate due to `rdfs:subPropertyOf` relations.

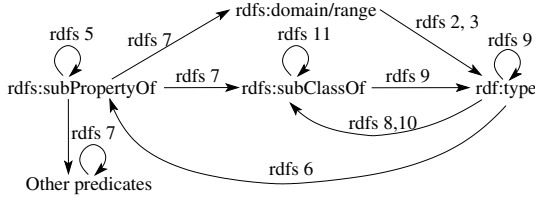


Fig. 1. Derivation Relationships among Predicates in The Rule Set

The self loops caused by `rdfs5`, `rdfs11`, `rdfs9` and `rdfs7` in the figure actually represent the problem P1. Without considering self loops, it is now clear from Fig.1 that `rdfs6`, `rdfs8`, `rdfs10` and R2 create big and complex loops in the derivation graph, which is just the cause of P2.

Since P2 is the main source of the inefficiency of the brute force method, now we first attack the problem P2. We find that, if some dependency relationships could be removed, then the graph could become a directed acyclic graph and the vertices (rules) could be topologically sorted, so the repetitive calculation in P2 can be reduced to just one pass. For this purpose, we now introduce the OSA (Original Semantics Assumption).

Original Semantics Assumption. Let \mathcal{A} be the set of RDF(S) axiomatic triples defined in the RDF Semantics document [3]. Let \mathcal{T} be the closure of \mathcal{A} under the rule set. Let Ω be the closure of the current RDF knowledge base under the rule set and $\mathcal{R} \subseteq \Omega$ be the set of all the triples whose subject is in the set $\{\text{rdfs:domain}, \text{rdfs:range}, \text{rdfs:type}, \text{rdfs:subPropertyOf}, \text{rdfs:subClassOf}\}$. The original semantics assumption supposes that $\mathcal{R} \subseteq \mathcal{T}$.

The assumption actually assumes that the RDF knowledge base does not strengthen the semantics of the five reserved RDF(S) property vocabularies.

They still keep their original meaning defined by the axiomatic facts. In most RDF(S) ontology engineering scenarios, this assumption is quite natural and can be satisfied because most applications does not require the change of the original RDF(S) semantics.

Under the OSA², the `rdfs6` and `rdfs10` rules

```
rdfs6: (uuu rdf:type rdf:Property) → (uuu rdfs:subPropertyOf uuu)
rdfs10: (uuu rdf:type rdfs:Class) → (uuu rdfs:subClassOf uuu)
```

become trivial. We can apply them only once after the `rdf:type` closure is obtained. We can be sure that no more triples can be obtained from the results of them. That is, they can not create loops in the derivation graph. Now let's look at the `rdfs8` rule:

```
rdfs8: (uuu rdf:type rdfs:Class) → (uuu rdfs:subClassOf rdfs:Resource)
```

Under the OSA, `rdfs:subClassOf` can not be other property's sub-property and `rdfs:Resource` can not be other class's sub-class. Therefore, this rule can be applied only once after the `rdf:type` closure is obtained. We can be sure that no more triples can be derived from the result of it. Hence this rule can not create loops in the derivation graph either. We can now safely remove the back lines caused by `rdfs6`, `rdfs8`, and `rdfs10` from Fig.1.

Finally, let's review R2. Because of the OSA, the five reserved vocabularies cannot be the sub property of other predicates. Hence the `rdfs7` rule can only create derivation relation from "other predicates" to the five reserved vocabularies in Fig.1. If R2 is now drawn on Fig.1, the only loop it can create is the one between the "other predicates" and the `rdfs:subPropertyOf`. However, this loop can be broken because the `rdfs:subPropertyOf` closure actually can be independently computed (step 2-4 of the algorithm described below).

Now, the only loops left in Fig.1 are the self loops. Except for these self loops, our algorithm can thus compute the entire closure in one pass.

In the following description, `?x`, `?y`, `?z` are used as variables to represent any resources or literals, with the restrictions that the predicates of a triple can only be URI references and the subjects of a triple cannot be literals.

1. Add all the RDF(S) axiomatic triples to the database.
2. Calculate the closure of all the triples with the form `(?x, rdfs:subPropertyOf, rdfs:subPropertyOf)`
 - This step can be done as follows. Let \mathcal{P} be a set of resources that initially has `rdfs:subPropertyOf` as the only member. For any triples of the form `(?x, \mathcal{P} , \mathcal{P})`, add `?x` to \mathcal{P} . Repeat this rule until fix point. Then \mathcal{P} should be the set of the possible sub properties of `rdfs:subPropertyOf`.
3. `(?x, \mathcal{P} , ?y) → (?x, rdfs:subPropertyOf, ?y)`
4. Calculate the `rdfs:subPropertyOf` closure, that is, `(?x, rdfs:subPropertyOf, ?y), (?y, rdfs:subPropertyOf, ?z) → (?x, rdfs:subPropertyOf, ?z)`
 - This step will be repeated until fix point.

² The Original Semantics Assumption is actually stronger than what we really needed. However, it is easier to explain and understand.

- After this step, we can obtain four sets \mathcal{D} , \mathcal{R} , \mathcal{C} and \mathcal{T} , which represent the set of the sub properties of the reserved property vocabularies `rdfs:domain`, `rdfs:range`, `rdfs:subClassOf` and `rdf:type` respectively.
- 5. $(?x, \mathcal{D}, ?y) \rightarrow (?x, \text{rdfs:domain}, ?y)$
- 6. $(?x, \mathcal{R}, ?y) \rightarrow (?x, \text{rdfs:range}, ?y)$
- 7. $(?x, \mathcal{C}, ?y) \rightarrow (?x, \text{rdfs:subClassOf}, ?y)$
- 8. Calculate the `rdfs:subClassOf` closure, that is, $(?x, \text{rdfs:subClassOf}, ?y)$, $(?y, \text{rdfs:subClassOf}, ?z) \rightarrow (?x, \text{rdfs:subClassOf}, ?z)$
 - This step will be repeated until fix point.
- 9. $(?x, \mathcal{T}, ?y) \rightarrow (?x, \text{rdf:type}, ?y)$
- 10. $(?x, ?y, ?z), (?y, \text{rdfs:domain}, ?a) \rightarrow (?x, \text{rdf:type}, ?a)$
- 11. $(?x, ?y, ?z), (?y, \text{rdfs:subPropertyOf}, ?a), (?a, \text{rdfs:domain}, ?b) \rightarrow (?x, \text{rdf:type}, ?b)$
- 12. $(?x, ?y, ?z), (?y, \text{rdfs:range}, ?a) \rightarrow (?z, \text{rdf:type}, ?a)$
- 13. $(?x, ?y, ?z), (?y, \text{rdfs:subPropertyOf}, ?a), (?a, \text{rdfs:range}, ?b) \rightarrow (?z, \text{rdf:type}, ?b)$
- 14. $(?x, ?y, ?z) \rightarrow (?x, \text{rdf:type}, \text{rdfs:Resource})$
- 15. $(?x, ?y, ?z) \rightarrow (?z, \text{rdf:type}, \text{rdfs:Resource})$
- 16. $(?x, ?y, ?z) \rightarrow (?y, \text{rdf:type}, \text{rdf:property})$
- 17. $(?x, \text{rdf:type}, ?y), (?y, \text{rdfs:subClassOf}, ?z) \rightarrow (?x, \text{rdf:type}, ?z)$
- 18. $(?x, \text{rdf:type}, \text{rdfs:Class}) \rightarrow (?x, \text{rdfs:subClassOf}, ?x)$
- 19. $(?x, \text{rdf:type}, \text{rdfs:Class}) \rightarrow (?x, \text{rdfs:subClassOf}, \text{rdfs:Resource})$
- 20. $(?x, \text{rdf:type}, \text{rdfs:Datatype}) \rightarrow (?x, \text{rdfs:subClassOf}, \text{rdfs:Literal})$
- 21. $(?x, \text{rdf:type}, \text{rdf:property}) \rightarrow (?x, \text{rdfs:subPropertyOf}, ?x)$
- 22. $(?x, ?y, ?z), (?y, \text{rdfs:subPropertyOf}, ?a) \rightarrow (?x, ?a, ?z)$

After applying these steps, it can be proved that under the OSA, all triples which can be entailed by the rule set have been added to the database. We have also verified the correctness of our algorithm using W3C RDF test cases.

Now let's turn to the problem P1. In the two steps of computing the transitive closure on `rdfs:subPropertyOf` and `rdfs:subClassOf`, actually a Floyd [4] like algorithm is used for better efficiency. Take `rdfs:subPropertyOf` for example. For each property `?y`, the rule “ $(?x, \text{rdfs:subPropertyOf}, ?y), (?y, \text{rdfs:subPropertyOf}, ?z) \rightarrow (?x, \text{rdfs:subPropertyOf}, ?z)$ ” is applied by table join to insert entailed triples into database in batches. This implementation has better performance than the two intuitive implementations as follows.

- One possible implementation is to apply the transitive rule directly on the database until fix point. On each iteration of applying the transitive rule, table join and batch insertion SQL command are used. The main drawback of such an implementation is that the less constrained self join of the `RDFSubClass` table will result in too many duplicate insertions, consuming too much time and memory when the data amount in the table is large.
- Another possible implementation is to read all the initial `rdfs:subClassOf` triples into the memory (if it is possible), perform the Floyd algorithm in the memory, and finally add the newly derived `rdfs:subClassOf` triples to the database. The main drawback of this implementation is that the one by one insertion of the resulting triples using SQL will consume much more time than using a similar batch insertion SQL command.

As to our Floyd like algorithm, the new triples are added to the database mainly in batches, while the self join of the `RDFSubClass` table is much more constrained, thus achieving better performance.

Finally, for the problem P3, on each rule application in our algorithm, we employ table join and batch insertion SQL command.

3.2 Inference in the Incremental Mode

In this section, we will describe the algorithm used in the incremental mode. In the incremental mode, every modification to the RDF(S) data will trigger computing the changes of the RDF(S) closure. The modification to the RDF(S) data can be regarded as appending some RDF triples and/or removing some RDF triples. We will discuss these two kinds of modification respectively.

When explicit triples are appended to the knowledge base, a simple forward-chaining closure computing is performed. The key point is that, according to the characteristic of the triple being appended, only a few rules in the rule set which may be relevant to the triple will be triggered. For example, if the predicate of the triple is not one of the RDF(S) reserved property vocabularies, only `rdf1`, `rdfs4a`, `rdfs4b` and `rdfs6` will be triggered.

When explicit triples are removed from the knowledge base, maintaining the consistency of the RDF(S) data is not as easy as when inserting triples. Since some triples derived from the removed triples may also be derived from some remaining triples, they could not be simply removed. However, examining one by one whether they can be derived from remaining triples is greatly time consuming. So we propose an algorithm which first removes all the suspect triples and then perform an incremental batch closure computing.

First, the triples to be removed are marked as `SUSPENDED` instead of being removed from the database immediately. Simultaneously, these triples are added to a queue for further processing.

Second, for each triple in the queue, the rules in the rule set that may be relevant to the triple are checked to see if there are `DERIVED` triples in the database that could be derived from the triple being processed.³ If such triples exist, they are marked as `SUSPENDED` and are added to the queue. This process continues until the queue becomes empty. Actually, it is a Breadth-First Search.

Third, all the `SUSPENDED` triples are removed from the database. All the tables which contain `SUSPENDED` triples are marked as `DIRTY`.

Finally, an incremental batch closure computing, similar to the algorithm used in the batch mode, is performed. The difference is that, if a table is not marked as `DIRTY`, the steps that add triples to that table will be skipped.

4 Query and Manipulation Through RQML

In order for both users and programs to query and manipulate RDF data in an uniform manner, here we define a declarative RDF Query and Manipulation

³ The triples that are marked as `SUSPENDED` are still regarded as valid triples in the database when the rules are being checked.

Language (RQML). RQML is designed based on several previous RDF query languages such as RQL[5], RDQL[2] and SeRQL[6].

The queries in RQML are designed to be maximally syntax-compatible with RDQL, while at the same time borrow features like path expression from SeRQL. In addition, as literal comparison and calculation are frequently used in practice, RQML provides direct support of these features on several widely used XML literals. Such support is not available in most of the previous RDF query languages.

In addition to the above features, RQML queries support sorting on the result URI references or literals. Further, the implementation of RQML queries allows the query results to be read in a streaming fashion. Therefore with all these features, the use of RQML in practice can be quite scalable.

Being also an RDF manipulation language in addition to an RDF query language, RQML includes some necessary manipulation commands like INSERT, DELETE and UPDATE.

Here are some examples of RQML commands.⁴

- Example of literal processing: Find all the employees who is older than 35 and taller than 1.75 meter and whose stature is greater than 0.08 times his/her age.

```
SELECT ?p WHERE      (?p, <!http://employee/age>, ?a),
                    (?p, <!http://employee/height>, ?h)
      SUCHTHAT ?a > 35 AND ?h. > 1.75 AND ?h. > 0.08 * ?a
```

- Example of UPDATE command: Update the level of all the engineer employees with Linux certification to Senior Engineer.

```
UPDATE (?p, <!http://employee/level>, ‘Senior Engineer’)
WHERE (?p, <!http://employee/level>, ‘Engineer’),
      (?p, <!http://employee/hasCert>, ‘Linux’).
```

5 Query and Inference Performance

In this section we report the results of the experiments performed to empirically evaluate the performance of query and inference of our approach. The inference performance test is first presented followed by the query performance test.

Two data sets are used in the experiments. One is an artificial data set called “T57” that consists of only `rdfs:subClassOf` and `rdfs:subPropertyOf` relation triples that construct a class hierarchy tree and a property hierarchy tree. Both the two trees have a maximum height of 7 and a constant fan-out of 5. Another data set is “WN”, which is the RDF representation of WordNet⁵. All experiments are performed on a PC with one Pentium-4 2.4GHz CPU and 1GB memory running Windows XP Pro, using J2SDK 1.4.1 and Eclipse-SDK-2.1.1 connecting to a local machine DB2 UDB V8.1 Workgroup Server. The inference

⁴ Please refer to <http://apex.sjtu.edu.cn/projects/orient/Documentation.htm#RQML> for more examples.

⁵ Available at <http://www.semanticweb.org/library/>

time is measured as the time cost to perform a full RDF(S) closure computing in the batch mode, starting from the database state of containing only the explicit triples. The query time is measured as the total time consumed from the sending of the query to finish fetching all the results from the database.

In our batch mode inference algorithm, the major component that determines the time complexity is the calculation of the transitive closure of sub properties and sub classes. The T57 data set is specifically designed to measure the empirical time complexity of it. By growing the two trees in T57 via adding one level of height each time, a series of growing data set for inference is got. The result is shown in Fig.2. The T57-1 line shows the relation between the inference time and the number of explicit triples. The T57-2 line shows the relation between the inference time and the number of triples after inference.

Different from T57, the WordNet data set has a very small RDF Schema with a very large set of instances and instance relations. The instance data in the four WordNet RDF files are sampled at the same speed. The number of sampled triples are multiplied by 5 each time and the triples from the four files are put together to get a series of growing data set for inference. The result is also shown in Fig.2. The WN-1 line shows the relation between the inference time and the number of explicit triples and the WN-2 line shows the relation between the inference time and the number of triples after inference.

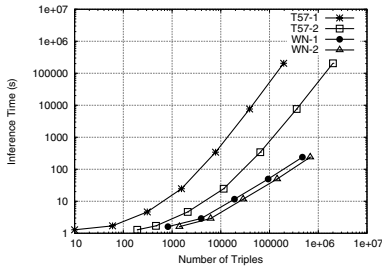


Fig. 2. RDF(S) Inference Performance

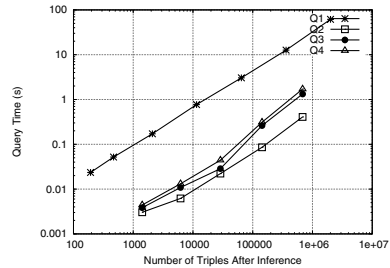


Fig. 3. RQML Query Performance

Both axes of Fig.2 are in log10 scale. The two T57 lines show a linear trend when the number of triples are large (> 1000). Linear regression analysis of the last four points at the end of each T57 line shows that the slopes are 1.87 and 1.75 for T57-1 and T57-2 respectively. This indicates approximately $O(n^{1.87})$ and $O(n^{1.75})$ time complexity. In theory, calculating transitive closure using Floyd algorithm has worst-case time complexity $O(n^3)$. When the algorithm is performed on a database, many factors of the RDBMS may further affect the performance. Combining the experiment result, we tend to empirically predicate that, when performed on a largely tree hierarchy ontology like T57 on a relational database, the time complexity of our batch inference algorithm is around $O(n^2)$.

Because the WordNet RDF data consists mostly of instance data, its number of inferred triples and inference time are much lower than T57. It, however, shows

the same trend of linear relation in Fig.2. Linear regression analysis of the last four data points at the end of each WN line indicates approximately $O(n^{0.92})$ and $O(n^{0.93})$ time complexity. Similarly, we empirically expect that, when performed on a largely instance data ontology like WordNet on a relational database, the time complexity of our batch inference algorithm is around $O(n)$.

We can see in Fig.2 that, in the experiments, the number of the triples after inference is in linear proportion to that of the explicit triples. If an RDF(S) ontology satisfies this property and the OSA, with characteristics between T57 and WN, we empirically estimate that its inference time complexity on a relational database is likely between $O(n)$ and $O(n^2)$. The n here can represent either the number of the explicit triples or the number of the triples after inference.

The RQML query performance is tested on both T57 and WordNet data set with derived triples. We used the following four queries to test sub-class query, simple query, query with join and query involving literals:

```
Q1: SELECT ?X WHERE (?X, <rdfs:subClassOf>, [aClass])
Q2: SELECT ?X WHERE (?X, <wn:similarTo>, [randomAdjective])
Q3: SELECT ?Y WHERE ([randomNoun] <wn:hyponymOf>, ?X),
      (?X, <wn:wordForm>, ?Y)
Q4: SELECT ?X WHERE (?X, <wn:wordForm>, ?Y) SUCHTHAT ?Y=[randomWordForm]
```

Q1 is performed on the T57 data set to obtain all subclasses of a given class. For each T57 data sample, and for each height of the tree hierarchy in that sample, Q1 is executed once using a class in that height. The query time of Q1 is then obtained as the average of the execution times. Q2, Q3 and Q4 are performed on the WordNet inferred data sets. The query time is averaged over 1000 query executions by randomly selecting a WordNet constant to replace the random constant in the above queries. The result is shown in Fig.3.

Both axes of Fig.3 is in log10 scale. Lines in Fig.3 are in linear trend, especially Q1. Linear regression analysis of the four lines shows approximately $O(n^{0.84})$, $O(n^{0.89})$, $O(n^{1.06})$, and $O(n^{1.05})$ time complexity for them. In theory, the worst-case time complexity of querying on one database table with indices is $O(n \log n)$. The actual query time also depends on the size of the result set. This test shows that the query time has a strong tendency of linear time $O(n)$ complexity and the query is executed quite speedy.

6 Related Work

There exist some other RDF(S) management systems such as Jena2 [2] and the Sesame system [6]. They also support storage and management of the RDF data on a relational database, but they only treat database as an alternative storage method. When answering queries, performing the inference on the fly is time consuming. In contrast, our approach performs efficient inference on database and stores all the derived triples to optimize the online query answering response.

The incremental inference algorithm used in our system is similar to the algorithm in [7]. The main difference is that, in [7], after marking some of the triples as **SUSPENDED**, these triples are examined whether they can be entailed

from explicit triples. However, as this examining process requires accessing the database very frequently and fragmentally, such an approach is not as efficient as it seems to be. On the other hand, although our algorithm may first remove some triples and then insert them back to the database, these removal and insertion operations are executed in a batch manner (that is, use less SQL commands to circumvent the problem P3 discussed in Sec.3). Thus, our algorithm could run faster than the one proposed by [7] in most cases.

Another inference algorithm that should be mentioned here is RETE [8]. As a generic rule-based forward-chaining algorithm, RETE is very efficient. However, as the RETE algorithm will consume lots of temporary memory in the inference procedure, it is not suitable for processing large-volume RDF data on databases.

7 Conclusion

In this paper, we present our approach on query, manipulation and inference of RDF data on relational databases. Different from most of the previous systems, the design of our approach aims at highly efficient query and inference with large-volume RDF data on relational databases. For query, we chose to store all the derived triples in database as well as the explicit ones. For inference, we carefully designed our algorithms for two inference modes, i.e. the batch mode and the incremental mode. We also defined a powerful RDF(S) query and manipulation language RQML, and presented the evaluation result of our approach.

References

1. Zhang, L., Yu, Y., Lu, J., Lin, C., Tu, K., Guo, M., Zhang, Z., Xie, G., Su, Z., Pan, Y.: ORIENT: Integrate ontology engineering into industry tooling environment. In: Proceedings of the 3rd International Semantic Web Conference (ISWC2004). (2004)
2. Wilkinson, K., Sayers, C., Kuno, H., Reynolds, D.: Efficient RDF storage and retrieval in Jena2. In: Proceedings of the first International Workshop on Semantic Web and Databases (SWDB), Berlin, Germany (2003) 131–150
3. Hayes, P., McBride, B.: RDF Semantics. W3C Recommendation, W3C (2004) <http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>.
4. Floyd, R.W.: Algorithm 97: Shortest path. Commun. ACM **5** (1962) 345
5. G.Karvounarakis, S.Alexaki, V.Christophides, D.Plexousakis, Scholl, M.: RQL: A declarative query language for RDF. In: Proceedings of the Eleventh International World Wide Web Conference (WWW02). (2002)
6. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying RDF and RDF Schema. In: Proceedings of the 1st International Semantic Web Conference (ISWC02). LNCS 2342 (2002) 54–68
7. Broekstra, J., Kampman, A.: Inferencing and truth maintenance in rdf schema. In: Proceedings of the First International Workshop on Practical and Scalable Semantic Systems (PSSS). (2003)
8. Forgy, C.: Rete: A fast algorithm for the many patterns/many objects match problem. Artificial Intelligence **19** (1982) 17–37

Clustering OWL Documents Based on Semantic Analysis

Mingxia Gao¹, Chunnian Liu¹, and Furong Chen²

¹ Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology, Beijing University of Technology, 100022, Beijing
gaomx@emails.bjut.edu.cn
ai@bjut.edu.cn

² R&D Center TravelSky Technology Limited
bjcfr@163.com

Abstract. Clustering OWL documents on the WWW or the Semantic Web is an important task in domain of ontology research and WI research. This paper analyzes semantic of OWL documents and proposes a method for computing semantic similarity between OWL documents. The method considers inheritance of objects and representation of complex classes. It can be used in clustering OWL documents built by experts and OWL documents learned by automatic tools.

1 Introduction

Ontology, special domain ontology [1, 2], plays an important role in information extract and exchange. These ontology documents must be sound and complete. Now, most of them had been built by experts. On the one hand this work consumes lots of time, on the other hand these ontologies have personal features. Clustering existing lots of ontology documents on the WWW or the Semantic Web is important for user to refine ontology or integrate ontology.

One of typical problems on Web Intelligence (WI) [3] technologies is PSML (Problem Solving Makeup Language). The core of PSML is distributed inference engines. The precondition of PSML is clustering appropriate contents and meta-knowledge like ontology information on the Semantic Web. Therefore clustering ontology documents on the WWW or the Semantic Web is very important for PSML.

OWL [4], which is the standard web ontology language proposed by W3C, has become the new standard for ontology representation and exchange on the Internet. It uses characteristic of other ontology languages for reference in its developing process. Clustering research in this paper aims at the OWL documents. Clustering other ontology languages like as: RDF, RDFS, DAML can amend the method in the paper.

1.1 Related Work

A key problem in clustering research is computing semantic similarity between objects. Traditional distance-based method in computing similarity between

database objects is not suitable for OWL documents. OWL essentially is semi-structure data. So evaluating semantic similarity can use methods of computing semantic similarity between XML documents for reference. The methods in literatures [5–8] can be divided into two kinds: one is structure similarity [5, 7, 8] and another is semantic similarity [6]. The common feature of structure similarity is modelling XML document as XML tree and evaluating similarity by tree operation [8] or path structure [5, 7]. Semantic similarity firstly computes similarity between basic elements in document, then evaluates full document based on these similarities. However, the methods can't directly evaluate similarity between OWL documents. The reason is that the method of structure similarity lacks semantic information whereas the method of semantic similarity only considers basic elements in XML document. OWL document is a language of representing knowledge. It can describe all kinds of objects in world and relations between the objects. The most important difference from XML is that OWL is an inferential language with semantic. It enhances inheritance between objects and complex classes representation.

This paper considers inheritance relation between objects and complex classes representation in OWL document. Then it proposes a method of computing semantic similarity between OWL documents and integrates it with hierarchical clustering algorithm to cluster OWL documents, which are built by experts or auto tools. The results of experiments show the algorithm has better effect on clustering OWL documents.

There are contributes in the paper as follow:

1. The paper proposes a method of computing similarity between simple classes based on resource similarity and property constraints.
2. When the paper evaluates similarity between complex classes, it uses set operation for reference.
3. The method in the paper has better effect on clustering OWL documents, which are built by experts or auto tools.

1.2 Paper Organization

The paper is organized as follows. Section 2 discourses upon method of computing similarity between classes in OWL documents. Section 3 introduces how to compute similarity matrix of OWL documents set. Experimental results are found in section 4. Section 5 concludes the paper and presents future work.

2 Similarity of Two Classes

An OWL concerns classes, properties, instances of classes (named individual). To compute similarity of two OWL documents, it is necessary to compute similarity of elements in OWL documents. Basic elements in OWL are classes. Classes have three types: simple named classes, anonymous classes, and complex classes. Commonly anonymous classes have not own local names, but they have properties which restrict instances of anonymous classes. Anonymous classes can be

seen as special simple classes. DatatypeProperty in OWL denotes relation between instances of classes and RDF literals or XML Schema and ObjectProperty denotes relation between instances of two classes. Similarity of properties including DatatypeProperty and ObjectProperty just explains similarity of two classes, which are domains of these properties. We propose a method to compute simple classes similarity that considers the basic semantic, properties of classes.

2.1 Similarity of Two Simple Classes

Members of simple classes generally are restricted by directly sup-classes and their properties as Figure 1. According to inheritance of class, a class can inherit its all sup-classes properties. In the way, restriction of sup-classes can translate into restriction of properties in sup-classes. In addition to consider basic semantic similarity of classes (names of simple named classes denote semantic), similarity between properties which restrict the classes in computing similarity of two classes must be considered.

Property can divided into two sorts: DatatypeProperty, ObjectProperty. Because two kinds of properties have different restrained rang, it is not meaning to compare similarity of different property types. So similarity of two classes can be computed by basic semantic similarity $BasicSim(c_1, c_2)$ (If c_1, c_2 have an anonymous class, then $BasicSim(c_1, c_2) = 0.$), similarity of DatatypeProperties in two classes, and similarity of ObjectProperties in two classes as Equation 1, where $w_1 = \frac{1}{|number(sum)|}$, $w_2 = \frac{|number(BMDP)|}{|number(sum)|}$, $w_3 = \frac{|number(BMOP)|}{|number(sum)|}$, $number(sum) = 1 + |number(BMDP)| + |number(BMOP)|$, $number(BMDP)$ and $number(BMOP)$ are number of the most mapping DatatypeProperty pair and number of the most mapping ObjectProperty pair respectively.

$$\begin{aligned}
 ClassSim(c_1, c_2) &= w_1 BasicSim(c_1, c_2) \\
 &+ w_2 \sum_{(p_i, p_j) \in BMDP} DataPropertySim(c_1.p_i, c_2.p_j) \\
 &+ w_3 \sum_{(p_i, p_j) \in BMOP} ObjectPropertySim(c_1.p_i, c_2.p_j)
 \end{aligned}
 \tag{1}$$

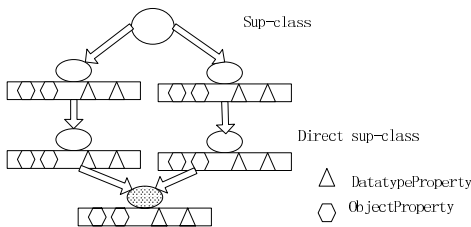


Fig. 1. Relation of class

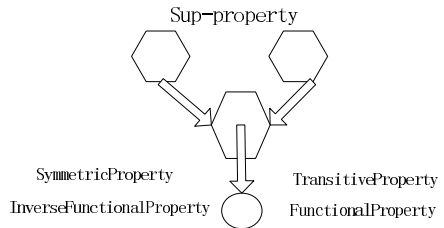


Fig. 2. ObjectProperty restriction

Classes, DatatypeProperties, ObjectProperties and individuals in OWL are essentially resources. Determining basic semantic similarity of these resources is matching their names. Their names include local names and namespaces. In general, different OWL documents have their own namespaces. Namespaces only denote location of resource. So computing similarity between names considers local name. Comparing rule is that if local names of two resources are same, value of their basic semantic similarity is "1"; otherwise value of their basic similarity is "0".

$DataPropertySim(p_1, p_2)$ denotes similarity of two DatatypeProperties. DatatypeProperties may range over RDF literals or simple types defined in accordance with XML Schema datatypes. If DatatypeProperty has rang restriction, determining similarity of DatatypeProperties must consider their rang restriction other than their basic semantic similarity as resources. Rang restriction directly compares Datatypes in Table 1, which are recommended for use with OWL. The similarity of DatatypeProperties is defined as Equation 2.

$$DataPropertySim(p_1, p_2) = \begin{cases} BasicSim(p_1, p_2) & \text{:no rang restriction} \\ w_1 BasicSim(p_1, p_2) + w_2 RangSim(p_1, p_2) & :w_1 + w_2 = 1 \end{cases} \quad (2)$$

Table 1. Datatype in OWL

xsd:string	xsd:normalizedString	xsd:boolean	xsd:decimal
xsd:float	xsd:double	xsd:integer	xsd:NCName
xsd:nonNegativeInteger	xsd:positiveInteger	xsd:nonPositiveInteger	xsd:negativeInteger
xsd:long	xsd:int	xsd:short	xsd:byte
xsd:unsignedLong	xsd:unsignedInt	xsd:unsignedShort	xsd:unsignedByte
xsd:hexBinary	xsd:base64Binary	xsd:dateTime	xsd:time
xsd:date	xsd:gYearMonth	xsd:gYear	xsd:gMonthDay
xsd:gDay	xsd:gMonth	xsd:anyURI	xsd:token
xsd:language	xsd:NMTOKEN	xsd:Name	

ObjectProperty relates individuals in two classes. An ObjectProperty can be defined to be a specialization (subproperty) of an existing ObjectProperty. In the condition, it can inherit super property's domain and rang as Figure 2. Similarity of ObjectProperty mainly is restricted by their ranges other than their basic similarity as resources. Though type restriction and cardinality restriction have a little influence on similarity of ObjectProperty, they can be neglected to avoid expensive time. So computing method of ObjectProperty similarity is the same as DatatypeProperty's (Equation 3).

However, similarity of ObjectProperty rang ($RangSim(p_1, p_2)$) is different from similarity of DatatypeProperty rang. It is mainly decided by similarity of classes which restricts rang of ObjectProperty. Note that if class of represent rang includes class of represent domain, the computing method of simple class

similarity can come forth infinity circulation. In the situation, similarity of ObjectProperty is to match their local names, that is basic semantic similarity.

$$ObjectPropertySim(p_1, p_2) = \begin{cases} BasicSim(p_1, p_2) & \text{no rang restriction} \\ w_1 BasicSim(p_1, p_2) & \text{: infinity circulation} \\ +w_2 ClassSim(p_1.c, p_2.c):w_1 + w_2 = 1 \end{cases} \quad (3)$$

To sum up, the detailed algorithm about similarity of two simple classes is in the appendix.

2.2 Similarity of Complex Classes

In order to improve reasoning capability, deducible ontology languages like as: DAML and OWL provide representation of complex classes comparing ontology language of metadata representation such as: RDF, RDFS. Representation of complex classes in OWL document includes Enumerated, Intersection, Union, Complement, and Equivalence. Complex classes also are sets. Evaluating their similarity can use similarity of two sets for reference.

Enumerated classes directly enumerate individuals when they are represented in OWL document. So similarity of two enumerated classes is the percent of number of same individuals in two classes on maximum number of individuals in every class. If there is only one Enumerated class in comparing two classes, similarity between them can directly use similarity of basic resource.

Example 1: Figure 3 represents three different classes "WineColor" in three OWL documents. According to the above comparing rule:

$$\begin{aligned} ClassSim(wine1.WineColor, wine2.WineColor) &= 2/3 \\ ClassSim(wine1.WineColor, wine3.WineColor) &= 1 \end{aligned}$$

Union, Intersection, Complement and Equivalence are set operations. Evaluating similarity between Union (Intersection, Complement, Equivalence) and simple class can be translated into computing similarity between members in set operations and simple class. We use theory of fuzzy sets to evaluate their similarity for reference. Table 2 shows detailed expression. Strictly speaking, the expression is not right. However, it can reflect value of similarity in a way. Note that if members of complex classes include complex classes again, the computing method will come forth infinite circulation. In the situation, comparing classes can directly be seen as resources and their similarity is their basic semantic similarity.

Example 2: Figure 4 is representation of complex classes in two OWL documents. According to computing rule:

$$ClassSim(WhiteWine, WhiteBurgundy) = \min \left\{ \begin{array}{l} ClassSim(Wine, WhiteBurgundy) \\ ClassSim(anonymity1, WhiteBurgundy) \end{array} \right\}$$

```

<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
    <owl:Thing rdf:about="#Red"/>
  </owl:oneOf>
</owl:Class>


---


<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
  </owl:oneOf>
</owl:Class>


---


<owl:Class rdf:ID="WineColor"/>

```

Fig. 3. Enumerated classes

```

<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>


---


<owl:Class rdf:ID="WhiteBurgundy"/>

```

Fig. 4. Intersection classes

Table 2. Computing method of set operations

Set operations	Logic	Computing equations
Union	$C = A \cup B$	$ClassSim(C, D) = \max\{ClassSim(A, D), ClassSim(B, D)\}$
Intersection	$C = A \cap B$	$ClassSim(C, D) = \min\{ClassSim(A, D), ClassSim(B, D)\}$
Complement	$C = -A$	$ClassSim(C, D) = 1 - ClassSim(A, D)$
Equivalence	$C = A$	$ClassSim(C, D) = ClassSim(A, D)$

3 Clustering OWL Documents

In addition to classes, similarity between instances is also considered because we call the set of individuals the extension of the class. An OWL document

can be parsed as set of classes and set of individuals. Similarity of two OWL documents is defined as weighted sum of classes sets similarity and individuals sets similarity.

If there are two sets of classes $\{c_1, c_2, \dots, c_n\}$ and $\{b_1, b_2, \dots, b_m\}$, they will give birth to the matrix of $m * n$, in which every element is similarity of corresponding two class in sets according to the algorithm in section 2. Classes sets similarity is the following equation.

$$ClassesSim(o_1, o_2) = \frac{\max\{\sum_{i=1}^n \max\{x_{i1}, \dots, x_{in}\}, \sum_{j=1}^m \max\{x_{1j}, \dots, x_{mj}\}\}}{\max\{n, m\}}$$

Similarity of individuals sets satisfies the following equation.

$$InstanceSim(o_1, o_2) = \frac{number(o_1, o_2)}{\max\{number(o_1), number(o_2)\}}$$

where $number(o_1, o_2)$ is number of same individuals in two OWL documents; $number(o_1)$ and $number(o_2)$ is number of individuals in documents o_1 and o_2 respectively. Generally, OWL documents built by experts represent vocabularies in domain and rarely deal with individuals. OWL documents learned by auto tools include lots of individuals. From the point a view, selecting weight in computing similarity of two OWL documents is different.

Set of n documents will obtain an $n * n$ matrix by two-to-two compare. The matrix is seen as similarity matrix in hierarchical clustering algorithm. However, we must select right weight, number of clusters, and threshold.

Generally, computing similarity of two OWL documents includes two circulations. The first is computing similarity of two classes. Complication of the algorithm is product of number of properties in two classes. The second is used to compute similarity of two documents. Complication of the algorithm is product of number of classes or individuals in two documents. Complication of computing similarity matrix is decided by number of documents and complication of computing similarity of two documents.

4 Experiments

To evaluate performance of the method in the paper, we collect two sets of OWL documents on different application domain from Internet. One is built manually; another is learned from text by auto tools. The number of classes and individuals in these documents ranges from 20 to 300. We implement the method in Java and run the experiments on 2.4GHZ Pentium 4 PC with 1GB RAM under Windows 2000 professional.

Table 3 shows the method of computing similarity matrix of OWL set has better effect on two kind sets of documents: OWL documents built manually and OWL documents learned by auto tools. However, efficiency of auto sets is better than manual sets'. Through random sampling, we find that there are fewer complex classes and relation of class-property in auto sets than in manual

Table 3. Process time

Number of documents	<i>Time of computing similarity matrix</i>	
	Auto set(s)	Manual(s)
10	45	78
20	190	369
50	1275	3010
200	39800	92000

Table 4. Documents sorts

Natural partitioning	Auto (30) Number	Natural partitioning	Manual (20) Number
University	5	Book	3
Trade	10	Person and animal	4
College	3	Travel	5
Publication	12	Food	8

sets. Computing similarity of two classes in manual sets expends more time than computing similarity of two classes in auto sets.

In order to evaluate precision of the algorithm, we select auto and manual documents that have clearly natural partitioning as Table 4. When setting right number of clusters, we obtain clusters that exactly match the original partitioning. However, we select different weight and threshold for auto sets and manual sets in clustering these documents. Manual building OWL documents generally illuminate vocabularies in domain and include a few individuals. So weight of individuals set is little. Auto learning OWL documents generally include lots of individuals. So weight of individuals set is more. In addition to, we select different threshold for auto sets and manual sets. The reason is that manual building OWL documents in different domain have less similarity than auto learning OWL documents from text. Though auto learning OWL documents belong to different domain, they have some same node as: root node. In the condition, similarity between these documents generally is not "0". So threshold of cluster is greater.

Besides studying performance and precision of the method in the paper, we also compared it with tree-edit distance in literature [7]. To our best knowledge, there are not methods which focus on computing similarity between OWL documents. This paper employs a method of computing similarity between XML documents—tree-edit distance to compute similarity between OWL documents. From the performance angle, complication of tree-edit distance is the same as the method in the paper. But practically, our experiments show that tree-edit distance algorithm has lower performance. The reason is that when parsing OWL document as normal XML tree, the number of nodes can be doubled. From the precision angle, when setting right number of clusters, we don't obtain clusters that exactly match the original partitioning. We find that some OWL documents without correlation obtain large value of similarity using tree-edit distance. The reason is that the method of tree-edit distance only considers structure. For ex-

Table 5. Two ObjectProperties in OWL

<code><owl:ObjectProperty rdf:ID="madeFromGrape"></code>	<code><owl:ObjectProperty rdf:ID="course"></code>
<code><rdf:domain rdf:resource="# Wine" /></code>	<code><rdf:domain rdf:resource="# Meal" /></code>
<code><rdf:range rdf:resource="WineGrape" /></code>	<code><rdf:range rdf:resource="MealCourse" /></code>
<code></owl:ObjectProperty></code>	<code></owl:ObjectProperty></code>

ample, there are two ObjectProperties in OWL documents as Table 5. Their semantic are not completely the same, but structure is the same to some extent.

5 Conclusion and Future Work

The paper proposes a method about computing similarity matrix of OWL documents set based on semantic analysis. In order to cluster OWL documents built by experts and auto tools, we integrate the method with hierarchical clustering algorithm. A great lot of experiments results show that the method has better effect on clustering OWL documents.

Refining or integrating ontologies is future research work according to clustering results. Another interesting work is management of ontology knowledge base.

Acknowledgements

The work is (Partially) supported by the NSFC major research program: "Basic Theory and Core Techniques of Non-Canonical Knowledge" (60496322) and Open Foundation of Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology.

References

1. Travis D. Breaux, Joel Reed. "Hierarchical Information Clustering Using Ontology Languages", In Proceedings of the 38 th Hawaii International Conference on System Sciences (HICSS-38), p. 112b, Honolulu, Hawaii, January 2005.
2. Brigitte Safar, Hassen Kefi. "Domain Ontology and Galois Lattice structure for Query Refinement" Proceedings of the 15th IEEE international conference on Tools with Artificial Intelligence, Computer Society 2003.
3. Zhong Ning, Liu Jiming, Yao Yiyu (Eds.) Web Intelligence, 440 p, Springer, 2003.
4. I. Horrocks, P. Patel-Schneider, F. Harmelen. "From SHIQ and RDF to OWL: the making of a Web Ontology language", Journal of Web Semantics, 1(1):7-26, 2003.
5. Wang Lian, David Wai-lok Cheung, Nikos Mamoulis, Siu-Ming Yiu. "An efficient and scalable algorithm for clustering XML Documents by structure". IEEE Transactions on Knowledge and Data Engineering, vol 16, 2004, pp82-96.
6. Mong Li Lee, Liang Huai Yang, Wynne Hsu, Xia Yang. "XCLust: Clustering XML schemas for effective integration", CIKM'02, McLean, Virginia, USA 2002.
7. J. Yoon, V. Raghavan, V. Chakilam. "BitCube: Clustering and Statistical Analysis for XML Documents", Thirteenth International Conference on Scientific and Statistical Database Management, Fairfax, Virginia, July 18-20, 2001.
8. Theodore Dalamagas, Tao Cheng, Klaas-Jan Winkel, Timos K. Sellis. "Clustering XML Documents by Structure". SETN 2004: 112-121. 108.

Appendix: The Detailed Algorithm

Algorithm: ClassSim

Input: Classes c_1, c_2 ;

Output: Class similarity

Step 1: Basic similarity

If $c_1 = c_2$ Then $BasicSim(c_1, c_2) = 1$; Else $BasicSim(c_1, c_2) = 0$

Step 2: computing similarity of DatatypeProperty

If there are DatatypeProperties in c_1 and c_2 Then $DPSim(c_1, c_2)$ Else $DPSim = 0$

Step 3: computing similarity of ObjectProperty

If there are ObjectProperties in c_1 and c_2 Then $OPSim(c_1, c_2)$ Else $OPSim = 0$

Step 4: similarity between simple classes

$ClassSim = w_1 BasicSim(c_1, c_2) + w_2 DPSim(c_1, c_2) + w_3 OPSim(c_1, c_2)$

Program: DPSim (The program of $OPSim(c_1, c_2)$ is similar as it.)

Input: Classes c_1, c_2

Output: Similarity sum

For every $p_{1i} \in DatatypeProperty(c_1)$

for every $p_{2j} \in DatatypeProperty(c_2)$

if p_1 and p_2 have not rang restriction $w_1 = 1, w_2 = 0$ else $w_1 = 0.5, w_2 = 0.5$

compute $DataPropertySim(p_{1i}, p_{2j})$

$SimMatrix = SimMatrix \cup (p_{1i}, p_{2j}, DataPropertySim)$

$MatchList = LocalMatch(SimMatrix, |DataProperty(c_1)|, |DataProperty(c_2)|)$

$DPSim = \frac{\sum_{DataPropertySim \in MatchList} DataPropertySim}{\max\{|DataProperty(c_1)|, |DataProperty(c_2)|\}}$

return DPSim

The following program finds the best matching pair of elements (DatatypeProperty or ObjectProperty). It will produce a one-to-one mapping, i.e., a DatatypeProperty in c_1 matches one DatatypeProperty in c_2 and vice versa.

Program: LocalMatch

Input: SimMatrix, m, n

Output: MatchList

MatchList={}

For $SimMatrix \neq \emptyset$ {

Select $(p_{1p}, p_{2q}, DataPropertySim)$ from SimMatrix condition as follow:

$DataPropertySim = \max_{(p_{1i}, p_{2j}, v) \in SimMatrix} \{v\}$

$MatchList = MatchList \cup \{(p_{1p}, p_{2q}, DataPropertySim)\}$

$SimMatrix = SimMatrix - \{(p_{1p}, p_{2j}, any) | (p_{1p}, p_{2j}, any) \in SimMatrix, j = 1, \dots, m\}$

$- \{(p_{1i}, p_{2q}, any) | (p_{1i}, p_{2q}, any) \in SimMatrix, i = 1, \dots, n\}$

}

return MatchList

An Ontology Based Approach to Construct Behaviors in Web Information Systems**

Lv-an Tang^{1,2}, Hongyan Li^{1,2,*}, Zhiyong Pan^{1,2}, Dongqing Yang²,
Meimei Li^{1,2}, Shiwei Tang^{1,2}, and Ying Ying^{1,2}

¹ National Laboratory on Machine Perception

² School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, P. R. China

{Tangla, Lihy, Panzy, Limm, Yingy}@cis.pku.edu.cn
ydq@db.pku.edu.cn
tsw@pku.edu.cn

Abstract. System behaviors specify the major functions of domain specific Web Information Systems (WIS). Traditional techniques can not satisfy various requirements or manage innumerable data while developing WIS behaviors. People appeal to a smart tool for implementing the WIS behaviors. This article makes the following contributions: (1) Proposes the concept of domain ontology and behavior ontology to describe the contents and operations of WIS; (2) Extends traditional ECA model to characterize the triggers, parameters, actions as well as validations of WIS behaviors; (3) Analyses the relationships between domain ontology and behavior ontology with rule sets; (4) Implements a tool named WISE Builder with four algorithms to help users building behavior ontology with domain ontology; (5) Shows the feasibility of this technique in a real application case.

Keywords: Web Information System, Behavior, Domain, Ontology, Code Generation.

1 Introduction

Featured with client logic, operational integration and high level knowledge management, WIS fetches information resources and carries out the business process by Web technologies^[1]. It is suitable for large applications such as ERP or Hospital Information Systems. However, associated with the application domains, WIS becomes both usage centralized and data intensive, various requirements and huge amount of data make WIS development a troublesome and complaining task.

From a user's view, the WIS can be roughly seen in two perspectives: (a) the static entities such as system structures, page presentations and the stored data, (b) the

** Supported by Natural Science Foundation of China (NSFC) under grant number 60473072.

* Corresponding author.

dynamic process such as data operations, system activities and interactions. They are quite different and hard to be integrated.

Researchers have proposed many methods such as MDA^[2], Hera^[3] to solve the problem. But in most cases, the users can not give requirements in detail and the designers may sketch wrong system frames due to misunderstanding domain knowledge^[4]. Some rough ideas of using ontology for building information systems can be found in reference [5] [6].

What is the advantage of using ontology in WIS development? The ontology helps to confirm user’s requirements and describes domain knowledge in both human understandable and computer process-able way. And the formalized ontology is also a good template for codes generation.

To describe the concepts of WIS clearly, two different ontologies: the domain ontology and behavior ontology are used in this paper (Figure 1). However, manually constructing them, especially the behavior ontology, is a time consuming, labor intensive and error prone task. The main reason is that behavior is just a process which can be viewed as a kind of logic or function codes, the invisibility makes it inherently difficult for modeling.

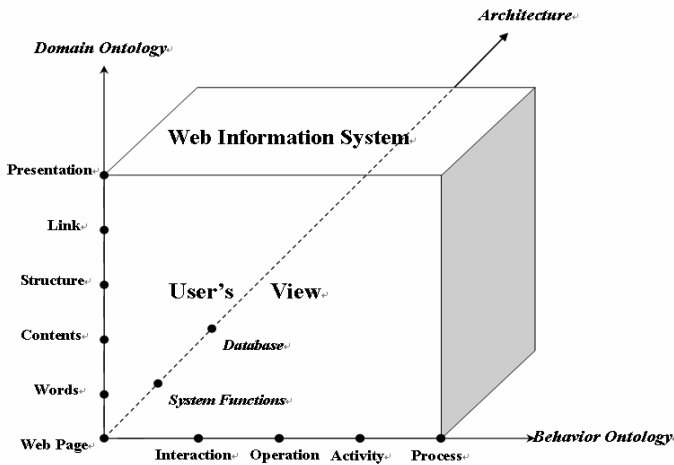


Fig. 1. Specify WIS from User’s View

To help users construct the behavior ontology easily, we extend traditional ECA model to characterize behaviors and implement an automatic tool named WISE Builder (WISE is short for *Web Information System auto-construction Environment*) with following features:

- WISE Builder implements the system behaviors as page elements and supports the “What You See Is What You Get” in ontology building;
- By analyzing the domain ontology, WISE Builder gets the characters of each page and generates the behavior ontology.

This article will discuss the models, techniques and tools for constructing both domain and behavior ontology. The rest of the paper is organized in the following manner: Section 2 presents some background knowledge and related work; Section 3 gives some concepts about WIS ontology, along with the problem specification; Section 4 gives the algorithms; Section 5 gives an application case; at last, Section 6 summarizes the paper and discusses future work.

2 Related Work

The WIS development is a hot research topic both in the industry and academia. Many projects are conducted by mapping the domain content to some logical function or physical representation. However, most of them pay little attention on the description and generation of WIS behaviors.

Model Driven Architecture (MDA)^[2] is a kind of software development style. The key point of MDA is to build a core model with high level abstraction independent on any implementation technology. With the help of mapping rules, the core model is transformed into one or more *Platform Specific Models(PSM)*, at last PSMs are transformed into codes. *MDA-based Approach for WIS Development(MIDAS)*^[7] is a MDA implementation for WIS development. It represents the whole system in structure model and defines a ring around to describe different platforms or support technologies. However, MIDAS does not specify the system behaviors in integrative style, the design and implementation about behavior models are remained as future work.

Scenario Based Design (SBD)^[8] defines system behaviors by describing how people use it to accomplish tasks and other activities. A user interaction scenario is a sketch of applications, which describes a sequence of events related in context including the goals, plans, and reactions. SBD is widely used to analyze software requirements, specify system's functionality and guide the design of interfaces. *Requirements Engineering Through Hypertext (RETH)*^[9] is a tool for requirements engineering through scenario. It presents a model that combines scenarios both with functions and goals. It also proposes a systematic and concrete design process that is both model driven and data driven. But those scenarios are stored in the format of tables or texts, which is hard to be parsed by other tools. And the scenarios may also be too prolific and deficient at the same time.

There are also some approaches based on ontology. *OntoWebber*^[10] is a part of the *OntoAgents* project developed by Stanford University. It integrates different aspects of Web sites, uses ontology as the foundation for sites design, and supports the reusable specifications of Web sites. *OntoWeaver*^[11] is also an ontology based approach to achieve high level support for Web site design and development. The *site ontology* is defined to provide fine-grained modeling support for user interfaces and Web site structures. Another similar tool--*WebRatio*^[12] has been presented on SIGMOD 2004. It extends a declarative language—*WebML* for specifying data intensive Web applications.

However, the models defined in those tools are too simple to describe the complex WIS behaviors. Mainly used to construct personal Web applications or normal Web

sites, they focus on the representation of data than manipulation. While WIS is enterprise oriented, which involves complex domain knowledge and business rules.

As our previous work, *Personalized tool for Ontology Development in Web Information System (PODWIS)*^[13] is an intelligent tool which provides graphical views in ontology building, discovers the frequent resources and composes them to a reusable component to improve efficiency. It is a pity that POWDIS does not support the modeling of WIS behaviors. To remedy this, a new approach called WISE Builder extended POWDIS to specify WIS behaviors. Table 1 gives a comparison of the above tools and WISE Builder.

Table 1. A Comparison of WISE Builder and Related Tools

	Target User	Data Integration	Reusability	Code Generation	Behavior Specification
MIDAS	Programmer	Strong	None	Strong	None
RETH	Domain User	None	None	None	Strong
OntoWebber	Domain User	Weak	Strong	Strong	None
OntoWeaver	Domain User	Weak	Weak	Strong	Weak
WebRatio	Designer	Strong	Strong	Strong	Weak
PODWIS	Domain User	Strong	Strong	Strong	None
WISE Builder	Domain User	Strong	Strong	Strong	Strong

3 Features in WIS Ontology

As a special context of WIS project, WIS ontology is made by two parts: the domain ontology and the behavior ontology. To get better compatibility, portability and integration, the WIS ontology is described in XML format. Thus it can be shared conveniently by other tools through XML interface.

To illustrate, we will derive the formal concepts of WIS ontology from *Web Health Resource Planning System (WHRP)*^[14], which is a large Web based hospital information system.

Example 1. The Web page of doctor diagnosis in WHRP is as Fig.2: The page can be analyzed according to two aspects: (a) the widgets, formats and framework illustrate domain content and system structure; (b) buttons such as ‘Add’, ‘Confirm’ specify the available operations and behaviors. Since the domain contents and system behaviors are rather complex. They will be defined step by step.

3.1 Review of Domain Ontology

The details of domain ontology in WIS development can be found in author’s previous work^[13]. The main concepts are as follows:

Doctor Diagnosis

Diagnosis Information

Type of Diagnosis: Outpatient Diagnosis

ICD Code: 16533 PinYin Code: Jin hua Description: Diagnosis for DD

Remarks: Other Diagnosis

Status: Others

MCR No.: N163 Diagnosed by: Han XiaoFeng

Sub-Specialty: 血液科 Diagnosed Date: 2005-03-02

Diagnosis Listing

Diag.Type	ICD10	Diagnosis Description	Remarks	Status	Diagnosed by	Sub-Specialty	Entered By	Entered On	Delete
Outpatient Diagnosis	16533	Diagnosis for RR		Main	Chen Ziqi	营养科		2005-03-02	<input checked="" type="checkbox"/>
Outpatient Diagnosis	16533	Diagnosis for LL		Others	Jia Junde	儿科		2005-03-02	<input type="checkbox"/>
Outpatient Diagnosis	16533	Diagnosis for DD	Other Diagnosis	Others	Wang DanHua	中医		2005-03-01	<input type="checkbox"/>
Outpatient Diagnosis	16533	Diagnosis for DD	Other Diagnosis	Others	Han XiaoFeng	血液科		2005-03-06	<input type="checkbox"/>

Fig. 2. The Page of Doctor Diagnosis

Definition 1. The basic item *I* of domain ontology is a 5-tuples, $I = \langle ItemName, ItemType, DisplayInfo, ValueInfo, Constrains \rangle$, where *DisplayInfo* is a set of display features used for the generation of Web pages, *ValueInfo* is a set of value’s source information, *Constraints* is a set of constraints for the value.

Note that, in Definition 1, all items in a page are distinguished by the unique *ItemName*; *ItemType* could be usual page widgets such as textfield, droplist, checkbox, etc; *DisplayInfo* includes item size, location and displayed text; *ValueInfo* illustrates the source of the item’s value such as database tables, sessions or user input streams; *Constraints* can be used to verify inputs.

Definition 2. The content piece *P* of domain ontology is a 3-tuples, $P = \langle PName, Elements, AddTag \rangle$, where *Elements* is a set which only contains other pieces or basic items, *AddTag* is a possible tag recording styles or shapes about the piece.

Note that, content pieces in WIS should be distinguished by names, *Elements* reflects the architecture of the piece, which only contains basic items or other pieces. *AddTag* records the style of the pieces.

3.2 The Definitions of Behavior Ontology

Although the content piece can record almost all user requirements, it can not specify the dynamic process of WIS in the following example.

Example 2. Consider the steps to arrange bed for a new patient: (1) login WHRP as a doctor, (2) go to patient admission page, (3) input patient and doctor’s information, (4) search an empty bed and arrange it for the patient. The whole process is described in scenario as Table 2.

Table 2. The Scenario of Bed Arrangement

User Behaviors	Interactions	WIS Behaviors	Web Page
1. Input user name and password	Click Button 'Login' → ← Go to WHRP's Main Page	<i>Check user name and password are not null.</i> Load the user information from database.	Log in
2. Choose the module to register patients	Click Button 'Patient Admission' → ← Go to Patient Admission Page	Read the user's information in session to see whether he has the authorization.	Main Page
3. Input the patient and his/her doctor's info.	Click Button 'Patient Admission' → ← Go to Bed Arrangement Page	<i>Check the inputs are validated.</i> Insert those input values to database tables	Patient Admission
4. Input the conditions such to query empty beds	Click Button 'Query Beds' → ← Update current page	<i>Check the conditions are validated.</i> Search the matched beds from database and display the result.	Bed Arrangement
5. Choose an empty bed in the result list	Click Button 'Arrange Beds' → ← The work is completed and back to Main Page	<i>Check whether a bed has been selected</i> Update the related data include the patient information and the bed status in different tables of database	Bed Arrangement

Although the structure and contents of Web pages can be represented in terms of basic items and pieces, the interactions, operations and process can not be described in domain ontology. To remedy this, we need to define some new concepts for the WIS behaviors.

The five WIS behaviors of Example 2 almost cover all types of user requirements for system functions in WHRP. We have following observations from the example:

Observation 1 (Trigger). The WIS behaviors are all triggered by user's actions on some basic items such as clicking buttons;

Observation 2 (Validation). Before behavior execution, the parameters must be validated to ensure the safety and efficiency;

Observation 3 (Key Action). Although behaviors are running in various and complex style, but their key operations can be reduced to simple actions such as insert, delete, etc;

Observation 4 (Fault Management). In most cases, behaviors can be carried out successfully, but there are times with errors, such as invalid password. Scenarios lack the ability to deal with this sort of situation, but behavior ontology should consider those things.

To format above observations, we introduce:

Definition 3. The Conditions C for behavior execution is defined as a set of 2-tuples in the format of

$C = \{(ItemName[k], ValidConstraints[k]) \mid k < N\}$ where N is the total number of involved basic items, $ItemName[k]$ is the item's name and $ValidConstraints[k]$ is a constraint about the item.

The behavior is executed if and only if all items satisfy corresponding constraint.

Definition 4. The Action A is defined as 3-tuples, $A = \langle ActionType, ParameterItemNames, Destination \rangle$, where $ParameterItemNames$ is a name set of the basic items whose values are the action's parameters and $Destination$ is used to store and display results.

Note that, in definition 4, $ActionType$ is one of the five frequent action types: 'Insert', 'Delete', 'Update', 'Select' and 'Clear'; Take the select action for example, $ParameterItemNames$ specifies the basic items whose values are the search conditions, item's $valueinfo$ provides the database table information and the $Destination$ is a list on the page to display search results;

Definition 5. The ResultLink R is defined as a 2-tuples, $R = \langle SuccessfulLink, ErrorLink \rangle$, where $SuccessfulLink$ is the link page if the behavior is completed, $ErrorLink$ is the one if the behavior ended unsuccessfully. The link can be the current page itself, thus the page will be updated to display results.

Based on the Observation 1-4 and definition 3-5, we finally give formal definition of behavior ontology:

Definition 6. The behavior B of WIS ontology is defined as a 5-tuples, $B = \langle BehaviorName, EventItem, Conditions, Action, ResultLink \rangle$, where $EventItem$ is a basic item on which the behavior is triggered, $Conditions$ is the execution condition in definition 3, $Action$ is the above definition 4, $ResultLink$ is the link page in definition 5.

Note that, in Definition 6, all behaviors in a Web page are distinguished by the unique $BehaviorName$, $EventItem$ is usually a button or selection in the page. An example of the behaviors is described in XML as in Figure 3.

To integrate the WIS functions with the structures and contents, we extended the domain piece to define the WIS ontology component

Definition 7. The Component C of WIS ontology is a 4-tuples, $C = \langle ComName, Elements, Behaviors, AddTag \rangle$, where $Behaviors$ is a set of behaviors defined on the component.

```

-<Behavior Name="Search Bed">
  -<EventItem>
    <Button> SearchBed </Button>
  </EventItem>
  - < Conditions >
    <Textfield Condition= "NotNull" >
      WardName </ Textfield >
    .....
  < /Conditions >
  - < Action >
    < ActionType>Select</ActionType>
    <ParameterItemNames >
    <Textfield> WardName </ Textfield >
    < Textfield > BedNO</ Textfield >
    .....
    </ParameterItemNames >
    <Destination>
      <Table>Bed Info</Table>
    </Destination>
  </Action>
  - < ResultLink>
    <SucessfulLink>Bed Arrangment</SucessfulLink >
    < ErrorLink >Search Error</ ErrorLink >
  < /ResultLink>
</ Behavior >

```

Fig. 3. The Behavior ‘Search Bed’

Note that, *Elements* reflect the component architecture, while *Behaviors* illustrates the functions and operations. This definition shows that a component could be applied to represent a whole interface page, or just a part of another component. The former one is called *Page Component*, and the latter one is called *Child Component*.

3.3 The Problem Specification

To make the construction of WIS ontology easier, we have implemented a visual tool for the non-professional users to build domain ontology and behavior ontology. However, in real practice we find that although a smart tool has been provided, the difficulty still exists in building behavior ontology: users are willing to build the domain ontology, because it is what the final WIS looks like--which is always in the user’s mind; the definition and formats of domain ontology are also clear at a glance. In contrast, the behavior ontology is more abstract and obscure, and the definition is too complicated to be manually built, especially the parts of *Conditions* and *Action*. Too many entries need to be filled in and users complained about that monotonous work.

Can we provide a tool to build the behavior ontology automatically or semi-automatically? A careful study on the features of domain and behavior ontology gives a new observation:

Observation 5. The WIS behavior's action types have correlations with the types of items in same component.

Example 3. There are 335 page components in WHRP's ontology, 232 components contain the basic item 'Table', all of them have the select action to search in the database. 155 components have to do insert to the database, 84.5% of them (131 components) contain more than 18 basic items.

Those curious situations can be explained by our development experience: the table item is mainly used to display search results, thus it is always connected with the select action. In the same way, if there are many textfields in the page, the input information should be stored in database, thus an insert behavior is needed. The experience can be represented as rule sets. To formalize the observation, we have following definition:

Definition 8. Let P be the piece of domain ontology and B be the corresponding behavior. The Action-Item Correlation rule is the expression in the format "IF X Then Y ", where X is a well-defined Boolean expression consisting of symbols in $\{\wedge, \vee\}$, Y is the action type of B .

With help of visual tools, users can define the rules from their own experiences and domain knowledge. Since the users prefer to build domain ontology rather than behavior ontology, why not generate the behaviors from the constructed domain ontology? Thus, an interesting problem arises and will be discussed in rest of this paper:

Problem Specification. Giving a constructed piece P of domain ontology, with the help of rule set R , generate the corresponding behavior B , and compose P with B to WIS ontology component C .

4 The Algorithms for WIS Behavior Generation

The behavior $B = \langle \text{BehaviorName}, \text{EventItem}, \text{Conditions}, \text{Action}, \text{ResultLink} \rangle$, where *BehaviorName*, *EventItem* and *ResultLink* can be built from default configurations, the *Conditions* can be composed by the basic items' constraints. The difficulty of behavior generation is building the *Action*, for it is the most complex part of the five, and the action type is also unknown. Our approach is: (a) determine the type, (b) generate action and other parts from given pieces of domain ontology, (c) assemble them to the behavior.

Algorithm 1 calculates the frequency of different items in pieces of domain ontology:

Algorithm 1 Statistics of Items

Input: A piece of domain ontology, named Piece ;

Output: A statistic set of piece items, named StatSet

Interior variables: a variable denotes the basic item, named Item;

Begin

1. Initialize the stack;
2. For each line of the file Piece, do
3. If the line contains '<' then
4. Push the current XML element in stack;
5. Else if the line contains '</' and the element.type != 'piece'
6. Pop stack top to variable Item;
7. If Item.Type is included in StatSet
8. Add 1 to 'Time' of corresponding;
9. Else
10. Create a new record R of StatSet;
11. $R.Type \leftarrow Item.type$;
12. $R.Time = 1$;
13. End if
14. End if
15. End for
16. Return StatSet;

END

Proposition 1. Let n be the total number of the piece elements, the complexity of Algorithm 1 is $O(n^2)$.

The proofs of the propositions are omitted here due to page limitation.

With the help of rule sets, we can determine the action types based on StatSet.

Algorithm 2 Determine Action Types

Input: The statistic set, named StatSet; the Action-Item correlation rule set, named RuleSet;

Output: The set of action types, named ActionTypes;

Interior variables: a variable denotes the rule in RuleSet, named Rule

Begin

1. Initialize the stack;
2. For each line of the file RuleSet, do
3. If the line contains '<Rule>'
4. Push the current XML element in stack;
5. Else if the line contains '</Rule>'
6. Pop stack top to variable Rule;

7. Search Rule. Presupposition in StatSet;
8. If(Rule. Presupposition in StatSet)
9. ActionTypes = ActionTypes \cup Rule.Result;
10. End if
11. End for
12. Return ActionTypes;

END

Proposition 2. Let n be the length of StatSet and m be the length of RuleSet, time complexity of Algorithm 2 is $O(m*n)$.

When types are confirmed, the action can be built. It is defined as 3-tuple $A = \langle \text{ActionType}, \text{ParameterItemNames}, \text{Destination} \rangle$, *ParameterItemNames* and *Destination* are concealed in the domain ontology. The following algorithm can generate them according to 5 different types of actions: *Insert*, *Delete*, *Update*, *Select*, *Clear*.

Algorithm 3 Building Actions

Input: The set of action types, named ActionTypes; the piece of domain ontology, named Piece;

Output: a set of generated actions, named ActionSet;

Interior variables: a variable denotes an action, named Action.

Begin

1. For each action type of the set ActionTypes, do
2. If the type is ‘Select’
3. Action \leftarrow Generate_Select_Action; //Procedure 1
4. ActionSet = ActionSet \cup Action ;
5. Continue;
6. Else if the type is ‘Insert’
7. Action \leftarrow Generate_Insert_Action; //Procedure 2
8. ActionSet = ActionSet \cup Action ;
9. Continue;
10.
11. Else if the type is ‘Clear’
12. Action \leftarrow Generate_Clear_Action; //Procedure 5
13. ActionSet = ActionSet \cup Action ;
14. Continue;

15. End if
16. End for
17. Return ActionSet;

END

We just present Procedure 1(generate the select action) here due to the page limitation.

Procedure 1 Generate_Select_Action

Input: The piece, named Piece;

Output: Generated select action, named SAction;

Interior variables: a variable denotes the basic item, named Item;
a variable denotes the table in database, named Table.

Begin

1. SAction. ActionType \leftarrow 'Select';
2. Search in Piece for the basic item 'table';
3. Item \leftarrow search result;
4. SAction. Destination \leftarrow Item.ItemName;
5. Table \leftarrow Item.ValueInfo.Table;
6. Search in Piece for the Basic Item whose ValueInfo.Table is the same as Table;
7. For each basic item I of the search result, do
8. Item \leftarrow I;
9. SAction. ParameterItemNames = ParameterItemNames \sqcup Item.ItemName;
10. End for
11. Return SAction;

END

Proposition 3. Let m be the size of ActionTypes set and n be the length of the piece, the complexity of Algorithm 3 is $O(m*n)$.

Algorithm 4 Generating Conditions

Input: a set of actions, named ActionSet; the piece of domain ontology, named Piece;

Output: a set of conditions, named ConditionSet;

Interior variables: a variable denotes the basic item, named Item; a variable denotes the action, named Action; a variable denotes the condition, named Condition;

Begin

1. For each action A in ActionSet, do
2. Action \leftarrow A;
3. For each ItemName in Action, do
4. Search in Piece for the basic item I whose name is the same as ItemName;
5. Item \leftarrow I ;
6. Condition.ItemName \leftarrow Item.ItemName;
7. Condition.ValidConstraints \leftarrow Item.Constraints;
8. ConditionSet = ConditionSet \cup Condition;
9. End for
10. End for
11. Return ConditionSet;

END

Proposition 4. Let a be the size of ActionSet, b be the size of the ParameterItemNames and c be the length of the piece, the complexity of Algorithm 4 is $O(a*b*c)$.

Note that, in practice, although b or c may be a large number, but the ActionSet's size is always small, so the total time complexity is acceptable.

Other parts of behaviors (BehaviorName, EventItem, ResultLink) can be generated from default settings instead of the domain ontology, because they are the interactions between users and computer, which are not strongly related to the structure and content of domain ontology. The default generated parts may not be exact, but they can be modified on a visual editor. After the five parts are generated, it is easy to compose them to the behaviors. The WIS ontology component is an integration of the domain piece and the generated behaviors. The algorithms for composition and integration are omitted here due to page limitation.

5 The Application of WISE Builder

The algorithms are implemented in an ontology development tool called *WISE Builder*. Non-professional users can use it easily to build WIS ontology. As it is a new exploration on WIS behavior generation, there is seldom similar existing algorithm to be compared with. Hence our experiment is combined with practical applications. The results show that the algorithms and WISE Builder work with acceptable speed and good quality.

WISE Builder is a part of the project *Web Information System auto-construction Environment (WISE)*^[15], which aims to construct WIS automatically from users' view, and achieve good extensibility and maintainability. They are currently used in developing a Web based hospital information system –WHRP, and they do a good job. There are totally 227 JSP pages and 108 HTML pages in WHRP, with 1,119 Java

Beans to contain the behavior functions, the system is also required to provide multi-views and support multi-formats. Using WISE Builder solved the following problems:

- Acquiring the requirements: The doctors and experts in hospital use WISE Builder to construct the ontology of WHRP so as to provide a precise requirements and system structure;
- Multi-view output: WHRP need to output a page component in many formats such as the JSP page for browsing or PDF page for printing, but the ontology building on WISE Builder was not affected, even they didn't need to know the existence of different views;
- Shorten developing period: The JSP pages and Java Beans are all generated from the corresponding page components in WHRP's ontology constructed by WISE Builder, system developers just need to adjust or custom some functions. Totally about 55% codes of the interface and 80% codes about the system behaviors are generated. The period of development is shortened greatly while the system is easier to be maintained.

6 Summary and Future Work

It is a new approach to automatically build WIS behaviors with domain ontology in WIS engineering. In order to carry out the WIS behaviors smartly, we have proposed an automatic tool named WISE Builder, the main contributions include:

- Providing graphical views to support the construction of behavior ontology;
- Extending ECA model to specify the trigger, data, destination and validation of WIS behaviors;
- Four algorithms for WIS behavior generation and four propositions about the algorithms;

The research of developing WIS based on ontology is just beginning, there is much more work to be done, such as:

- Implementing user custom action and complex behaviors: WISE Builder can auto build the frequent actions, but the user custom action should also be carried out due to various situations;
- Refining the generated codes: There are also redundancies in the generated WIS codes. Reduce redundancy and increase efficiency is a main task of our future work.

Acknowledgments

Several people have contributed to the project and this paper, they are: Ming Xue (IBM China), Baojun Qiu, Jianjun Wang, Lei Wang, Bin Zhou, Ke Li, Mengqing Wu, etc.

We would like to thank for their helpful discussions and we also want to thank four anonymous reviewers for their comments.

References

1. Tomas Isakowitz, Michael Bieber, Fabio Vitali: Web Information Systems. Communications of the ACM, Page: 78 - 80, Vol 41(7), ACM Press, July 1998;
2. Joaquin Miller and Jishnu Mukerji. (Eds). Model Driven Architecture—The MDA Guide, OMG, 2003. Retrieved from: <http://www.omg.com/mda>, 2003;
3. Habib, Reza, Nyberg, Lars, Tulving, Endel: Hemispheric asymmetries of memory: the HERA model revisited. Trends in Cognitive Sciences Volume: 7, Issue: 6, June, 2003, pp. 241-245;
4. Stefano Ceri, Ioana Manolescu: Constructing and integrating data-centric Web Applications: Methods, Tools, and Techniques. In Proceedings of VLDB 2003, Berlin, Germany ;
5. Kietz, J., Maedche, A, etc: Extracting a Domain-Specific Ontology Learning from a Corporate Intranet. In Proceedings of Learning Language In Logic Workshop 2000, Lisbon, Portugal;
6. Ding, Y., & Foo, S.: Ontology research and development. Journal of Information Science, Vol 28(5), 375-388, 2002;
7. Paloma Caceres, Esperanza Marcos, Belen Vela: A MDA-Based Approach for Web Information System Development. In Proceedings of WISME 2003, October 21st, 2003, San Francisco, USA;
8. Mary Beth Rosson, John M.Carroll: The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. Lawrence Erlbaum Associates, 2002;
9. Hermann Kaindl: Active Tool Support for Requirements Engineering Through RETH. In Proceedings of the 12th IEEE International Requirements Engineering Conference(RE 2004), September 6-10, 2004, Kyoto, Japan;
10. Yuhui Jin, Sichun Xu, Stefan Decker: OntoWebber: A Novel Approach for Managing Data on the Web. In Proceedings of EDBT 2002, Prague, Czech Republic;
11. Y. Lei, E. Motta, and J. Domingue: Modelling Data-Intensive Web Sites with OntoWeaver. In Proceedings of WISM 2004, Riga, Latvia, 2004;
12. Marco Brambilla, Stefano Ceri, etc: Declarative Specification of Web Applications exploiting Web Services and Workflows. In Proceedings of the ACM SIGMOD 2004, Paris, France;
13. Lv-an Tang, Hongyan Li, etc: PODWIS: A Personalized Tool for Ontology Developing in Domain Specific Web Information System. In Proceedings of APWeb 2005, Shanghai, China;
14. Hongyan Li, Ming Xue, Ying Ying: A Web-based and Integrated Hospital Information System. In Proceedings of IDEAS04-DH, September 29-31, 2004, China;
15. Ming Xue, Hongyan Li: Managing User Interaction Forms on Web Pages: A Component-base approach. Journal (Natural Science) Of Peking University, Vol. 40(3), May 2004, pp 473~479;

A Semi-automatic Ontology Acquisition Method for the Semantic Web

Man Li, Xiaoyong Du, and Shan Wang

School of Information, Renmin University of China, Beijing 100872, China
liman1@ruc.edu.cn

Abstract. The success of the Semantic Web strongly depends on the proliferation of ontologies, which requires fast and easy engineering of ontologies. The paper analyzes the semantic similarity between relational model and ontology, and proposes a semi-automatic ontology acquisition method(SOAM) based on data in relational database. SOAM tries to ensure the quality of constructed ontology and the automatic degree of acquiring process by balancing the cooperation between user contributions and machine learning. Because OWL is the latest ontology language standard recommended by W3C, the implementation of SOAM is given to acquire OWL ontology automatically as much as possible. Different from existing methods, the implementation method not only can acquire OWL ontology from relational database directly without demanding a middle model, but also can refine obtained ontology according to existing lexical knowledge repositories semi-automatically.

1 Introduction

The Semantic Web[1] proposed by Tim Berners-Lee has been regarded as the next generation of the current Web, which aims to add semantics and better structure to the information available on the Web. With the development of Semantic Web research, people have realized that the success of Semantic Web depends on the proliferation of ontologies and pay more attention to the construction of ontologies. Though ontology construction tools have become mature over the last decade, the manual development of ontologies still remains a tedious and cumbersome task. So this paper concerns how to acquire ontology automatically or semi-automatically from existing resources.

Due to the wide use of relational database in information management, a large amount of data about various domains are organized and stored in relational database. Data in relational database may be used as a kind of important resource for acquiring ontology. The paper proposes a semi-automatic ontology acquisition method(SOAM) based on data in relational database by analyzing the semantic similarity between relational model and ontology. SOAM divides ontology acquisition process into two stages and proposes the necessity of refining the obtained ontological structure before acquiring ontological instances. SOAM tries to balance the cooperation between user contributions and machine learning reasonably to ensure the quality of constructed ontology and the automatic degree of acquiring process.

OWL[2] is the latest standard recommended by W3C. It facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics. So in this paper the implementation of SOAM is introduced to acquire OWL ontology from data in relational database. Different from existing methods, the implementation method not only can acquire OWL ontology from relational database directly by using a group of rules without demanding a middle model, but also can refine obtained ontology according to existing lexical knowledge repositories semi-automatically.

The paper is organized as followings. Section 2 analyzes the semantic similarity between relational model and ontology and then proposes the SOAM . Section 3 introduces the implementation of SOAM in detail. Section 4 gives a case study. Section 5 introduces related works. Section 6 draws a conclusion and gives the future works.

2 Acquiring Ontology from Relational Database

To explain the semi-automatic ontology acquisition method based on data in relational database clearly, the semantic similarity between relational model and ontology are analyzed firstly here.

2.1 Semantic Similarity Between Relational Model and Ontology

The underlying model of relational database is the relational model[3]. The model includes a finite set R called relations, a finite set A called attributes, primary keys and foreign keys etc. Some functions in common use are as following.

- A function $dom(A_i)$. It acquires the value's range of attribute A_i , where $A_i \in A$.
- A function $attr(R_i)$. It acquires the attributes contained in a specific relation R_i , where $R_i \in R$ and $attr(R_i) \subseteq A$.
- A function $pkey(R_i)$. It acquires the primary keys in a given relation R_i , where $R_i \in R$ and $pkey(R_i) \subseteq attr(R_i)$.
- A function $fkey(R_i)$. It acquires the foreign keys in a given relation R_i , where $R_i \in R$ and $fkey(R_i) \subseteq attr(R_i)$.

Besides the above entities, relational model includes some further constraints, such as NOT NULL, UNIQUE etc, and some dependencies. All these are called relation schema, which is used to describe the structure and association of data. The tuples in relations reflect the values of schema, and they are content of database.

Ontological structure is a 5-tuple $O := \{C, R, H^c, rel, A^o\}$ [4].

- A finite set C . It is called *concepts*.
- A finite set R . It is called *relations*.
- A concept hierarchy H^c . H^c is a directed relation $H^c \subseteq C \times C$, which is called concept hierarchy or taxonomy. For example, $H^c(C_1, C_2)$ specifies that C_1 is a subconcept of C_2 .

- A function $rel : R \rightarrow C \times C$. It relates concepts non-taxonomically. For example, $rel(R) = (C_1, C_2)$ specifies that C_1 and C_2 have relation R.
- A set of ontological axioms A^o . A^o is expressed in an appropriate logical language, e.g. first order logic.

Based on the ontological structure, ontology comprises a set of instances, which could be seen as the extension of concepts.

Both relational model and ontology are a kind of model for organizing knowledge, and there are some semantic similarities between them. In relational model, both entities and relationship of entities are expressed by relations. Therefore one relation in relational model may be corresponding to an ontological concept or relation. If two relations in database have inheritance relationship, then the two corresponding ontological concepts(or relations) will have hierarchical relationship. Additionally, attribute constraints in database may be converted to ontological axioms, and the tuples in database may constitute ontological instances. The situations are the simplest semantic relation between relational database and ontology. In fact there are more complicated cases. For example, the information spread across several relations may need to be integrated into one ontological concept, and so on.

From the above analysis, it can be seen that there exists some semantic similarity between relational model and ontology. In general, the relational database schema is corresponding to ontological structure and the tuples in database are corresponding to ontological instances, which is shown in Fig.1.

So it is feasible to acquiring ontology from relational database according to the semantic similarity between relational model and ontology.

2.2 SOAM

According to Fig. 1, the process of acquiring ontology can be divided into two stages: acquiring ontological structure and acquiring ontological instances. In the stage of acquiring ontological structure, it is necessary to capture information about database schema firstly, and then based on the information ontological structure can be constructed. Since the constructed ontological structure may not be ideal, the evaluation and refinement about it is needed. After that, a more satisfying ontological structure is formed. Whereafter the stage of acquiring ontological instances can start. So we propose the semi-automatic ontology acquisition method, SOAM, which consists of four steps.

- Step 1.** Capture the information about relational database schema;
- Step 2.** Acquire ontological structure according to the database schema information;
- Step 3.** Refine the obtained ontological structure;
- Step 4.** Acquire ontological instances based on refined ontological structure.

Step 1 captures the information about relations, attributes, primary keys, and foreign keys, etc, in relational database. Based on Step 1, Step 2 acquires ontological concepts, organizes them into the taxonomy and detects non-taxonomical

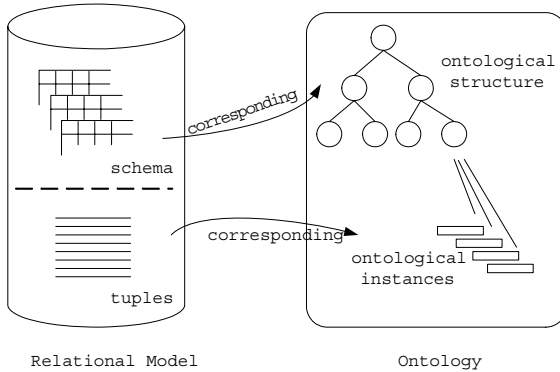


Fig. 1. Semantic Similarity Between Relational Model and Ontology

relation between two concepts and ontological axioms. Step 3 lets user validate the obtained ontological structures and form the final result. The step is necessary, because ontological structure is the backbone of ontology. The cost of ontological structure's change is large because it will affect a large number of related instances. So we let user evaluate and refine the obtained ontological structure. After step 3, a steadier and more satisfying ontological structure is formed. Then step 4 can be implemented easily based on the ontological structure.

In the four steps, Step1, Step 2 and Step 4 can be implemented automatically. Step 3 needs users' intervention. The whole process is semi-automatic. SOAM tries to balance the cooperation between user contributions and machine learning in order to ensure the quality of constructed ontology and improve the automatic degree of acquiring process.

3 Implementation of SOAM

In fact, although all the ontology description languages embody the basic ontological structure, they have different characteristics and expressive ability. So it is necessary to choose appropriate language for ontology construction. Here, the reason of choosing OWL is explained and the characteristics of OWL is introduced firstly. Then the implementation of the steps in SOAM is introduced to acquire OWL ontology. Step 1 can be implemented easily by reverse engineering of relational database[5], so it will not be discussed in this paper. Step 2, Step 3 and Step 4 are important in SOAM. The implementation of them will be given in detail.

3.1 Choice of Ontology Description Languages

Up to now, there are many ontology languages. The earlier languages are used to describe ontologies for specific user communities (particularly in the sciences

and in company-specific e-commerce applications), they are not defined to be compatible with the architecture of the World Wide Web in general, and the Semantic Web in particular. Recently, with the development of Web, some Web-based ontology description languages, such as SHOE, RDF(S), DAML+OIL and OWL are proposed. Among these languages, OWL is the latest standard recommended by W3C for the Semantic Web. OWL uses both URIs for naming and the description framework for the Web provided by RDF to add the following capabilities to ontologies.

- Ability to be distributed across many systems
- Scalability to Web needs
- Compatibility with Web standards for accessibility and internationalization
- Openness and extensibility

OWL is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics. So we choose OWL as ontology description language.

OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. Most of the elements of an OWL ontology concern classes, properties and instances. The members of a class are known as instances of the class. Properties are used to assert general facts about the members of classes and specific facts about instances. OWL distinguishes two types of properties: datatype properties and object properties. Datatype properties are relations between instances of classes to datatypes. Object properties are relations between instances of two classes. In addition, property characteristics are used to further specify properties. Both properties and classes can be arranged in a hierarchy.

3.2 Acquiring Ontological Structure

In SOAM, step 2, viz. acquiring ontological structure is a key step. The paper assumes that the source relational schema is at least in third normal form (3NF). The assumption does not limit the applicability of our method, because most of databases are highly likely to satisfy the 3NF requirement and there is algorithm that can convert 1NF to 3NF easily[6]. We give the related definitions firstly.

Definition 1. For relations R_i and R_j in database, suppose that $A_i \subseteq attr(R_i)$ and $A_j \subseteq attr(R_j)$ hold, $t_i(A_i)$ expresses the values of tuple t_i in attributes A_i and $t_j(A_j)$ expresses the values of tuple t_j in attributes A_j . For each $t_i(A_i)$ in R_i , if in R_j there exists $t_j(A_j) = t_i(A_i)$, A_i and A_j are called inclusion dependency, denoted as $R_i(A_i) \subseteq R_j(A_j)$.

Definition 2. For relations R_i and R_j in database, suppose that $A_i \subseteq attr(R_i)$ and $A_j \subseteq attr(R_j)$ hold. If there exist $R_i(A_i) \subseteq R_j(A_j)$ and $R_j(A_j) \subseteq R_i(A_i)$, A_i and A_j are called equivalence, denoted as $R_i(A_i) = R_j(A_j)$.

The rules for acquiring ontological structure are given below.

Rule 1. For relations R_1, R_2, \dots, R_i in database, suppose that $A_1 = pkey(R_1)$, $A_2 = pkey(R_2), \dots, A_i = key(R_i)$. The information spread across R_1, R_2, \dots, R_i

should be integrated into a class C_i in ontology, if $R_1(A_1) = R_2(A_2) = \dots = R_i(A_i)$ is satisfied.

Comment: Rule 1 indicates that if several relations are used to describe one entity, the information in them should be integrated into one class.

Rule 2. The relation R_i can be mapped to an ontological class C_i , if rule 1 cannot be applied, and one of the following conditions can be satisfied: (i) $|pkey(R_i)| = 1$; (ii) $|pkey(R_i)| > 1$, and there exists A_i , where $A_i \in pkey(R_i)$ and $A_i \notin fkey(R_i)$.

Comment: Rule 2 indicates that if one relation is used to describe an entity, but not relationship between relations, then it can be mapped into one ontological class.

Rule 3. For relations R_i and R_j , $A_i(A_i \subset attr(R_i))$ can be mapped to an ontological object property P , if $A_i \not\subset pkey(R_i)$ and $R_i(A_i) \subseteq R_j(A_j)$ are satisfied. Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively, and then the domain and range of P are C_i and C_j .

Comment: Rule 3 indicates that if one attribute only reflects reference relationship between two relations, it can be used as an object property in ontology.

Rule 4. For relations R_i and R_j , two ontological objects property “has-part” and “is-part-of” can be created, if the two conditions are satisfied: (i) $|pkey(R_i)| > 1$; (ii) $fkey(R_i) \subset pkey(R_i)$, where $fkey(R_i)$ referring to R_j . Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively. The domain and range of “is-part-of” are C_i and C_j and the domain and range of “has-part” are C_j and C_i . Properties “has-part” and “is-part-of” are two inverse properties.

Comment: Rule 4 indicates that if the entity described by one relation is a special kind of the entity described by the other, then the classes corresponding to the two relations should have objects properties “is-part-of” and “has-part” respectively.

Rule 5. For relations R_i , R_j and R_k , two object properties P'_j and P''_j can be created based on the semantics of R_k , if $A_i = pkey(R_i)$, $A_j = pkey(R_j)$, $A_i \cup A_j = fkey(R_k)$ and $A_i \cap A_j = \emptyset$ are satisfied. Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively. The domain and range of P'_j are C_i and C_j and the domain and range of P''_j are C_j and C_i . P'_j and P''_j are two inverse properties.

Comment: Rule 5 indicates that if one relation is used to describe the relationship between two relations, two inverse object properties can be acquired based on semantics of the relation.

Rule 6. For relation R_1, R_2, \dots, R_i and R_j , object properties $P_i^1, P_i^2, \dots, P_i^{i*(i-1)}$ can be created and the semantics of them are based on the decomposition of n-ary relationship provided by R_j , if $A_1 = pkey(R_1)$, $A_2 = pkey(R_2)$,

$\dots, A_i = pkey(R_i), A_1 \cup A_2 \cup \dots \cup A_i = fkey(R_j)$ and $A_1 \cap A_2 \cap \dots \cap A_i = \phi$ are satisfied.

Comment: Rule 6 indicates that if one relation is used to describe the relationship among multi-relations, it should be decomposed to multiple object properties in ontology.

Rule 7. For an ontological class C_i and the datatype properties set of C_i denoted as $DP(C_i)$, suppose that C_i is corresponding to relations R_1, R_2, \dots, R_i in database. For every attribute in R_1, R_2, \dots, R_i , if it cannot be mapped to object property by using rule 3, then it can be mapped into ontological datatype property of C_i . The domain and range of each property P_i are C_i and $dom(A_i)$ respectively, where $P_i \in DP(C_i)$ and $A_i \in attr(R_i)$.

Comment: Rule 7 indicates that for each attribute in relations, if it cannot be converted into ontological object property, it can be converted into ontological datatype property.

Rule 8. For relation R_i and R_j , suppose that $A_i = pkey(R_i), A_j = pkey(R_j)$. If rule 1 does not applied and $R_i(A_i) \subseteq R_j(A_j)$ is satisfied, then the class/property corresponding to R_i is a subclass/subproperty of the class/property corresponding to R_j .

Comment: Rule 8 indicates that if two relations in database have inheritance relationship, then the two corresponding ontological class or property can be organized in a hierarchy.

Rule 9. For relation R_i and $A_i \in attr(R_i)$, the minCardinality and maxCardinality of the property P_i corresponding to A_i are 1, if $A_i = pkey(R_i)$ or $A_i = fkey(R_i)$ is satisfied.

Rule 10. For relation R_i and $A_i \in attr(R_i)$, the minCardinality of the property P_i corresponding to A_i is 1, if A_i is declared as NOT NULL.

Rule 11. For relation R_i and $A_i \in attr(R_i)$, the maxCardinality of the property P_i corresponding to A_i is 1, if A_i is declared as UNIQUE.

Comment: Rule 9, 10 and 11 indicate that some constraints of attribute in relation may be converted to the property cardinality in ontology.

According to Rule 1 to Rule 11, an OWL ontological structure can be acquired from a relational model.

3.3 Refining Ontological Structures

Based on the above rules, ontological structure will be acquired automatically. However, the quality of the obtained ontological structure lies on the quality of data in the selected database. In the most circumstances, the obtained ontological structure is coarse. In addition, some semantics of obtained information need to be validated. So step 3 in SOAM, viz. refining obtained ontological structure is necessary.

Because existing repositories of lexical knowledge usually includes authoritative knowledge about some domains, the paper suggests refining obtained ontology according to them, especially machine-readable dictionaries and thesauri. For example, HowNet, WordNet and some domain thesauri can be used as the references for refining ontology.

The refinement algorithms is shown in Algorithm 1. The basic idea is that when a user wants to refine a concept in the ontology, the algorithm can help him find some similar lexical entries in the existing lexical repositories. Then the user can refine the ontological concept according to the similar lexical entries' information, such as hyponym, antonym and synonym etc.

Algorithm 1. (Refinement algorithm)

Input: an ontological concept C waiting to be refined, a machine-readable lexical repositories LR , similarity measures sim , return threshold k

Output: reference lexical entries

Begin

for each lexical entry $le \in LR$ do

 compute $sim(C, le)$;

return the k most similar lexical entries to the user.

End

In algorithm 1, the choice of similarity measures is important. The simplest measure is lexical similarity. Because the *edit distance* [8] formulated by Levenshtein is a well-established method for weighing the difference between two strings. For two concepts L' and L'' , the lexical similarity measure $LSim(L', L'')$ can be given based on *edit distance*, which is shown in formula (1).

$$LSim(L', L'') = \max(0, \frac{\min(|L'|, |L''|) - ed(L', L'')}{\min(|L'|, |L''|)}) \in [0, 1],$$

where $ed(L', L'')$ is the *edit distance* between L' and L'' . (1)

$LSim$ returns a degree of similarity between 0 and 1, where 1 stands for perfect match and 0 for bad match. It considers the number of changes that must changes against the length of the shortest string of these two.

However, lexical similarity measure is not sufficient. The paper gives the similarity measure in conceptual level, which considers the similarity about super-concepts ($Sup(L')$ and $Sup(L'')$) and sub-concepts ($Sub(L')$ and $sub(L'')$) of compared two concepts L', L'' .

The similarity about super-concepts of two concepts L' and L'' is denoted as $SupSim(L', L'')$, which is shown as formula (2).

$$SupSim(L', L'') = \frac{\sum_{j=1}^n \max_{i=1}^m (LSim(L_i, L_j))}{\max(|Sup(L')|, |Sup(L'')|)},$$

where $L_i \in Sup(L'), L_j \in Sup(L'')$. (2)

For two concepts L' and L'' , formula (2) summarizes the lexical similarity of each pair super-concepts of them against the most super-concepts of these two.

The similarity about sub-concepts of two concepts L' and L'' is denoted as $SubSim(L', L'')$, which is shown as formula (3).

$$SubSim(L', L'') = \frac{\sum_{j=1}^n \max_{i=1}^m (LSim(L_i, L_j))}{\max(|Sub(L')|, |Sub(L'')|)},$$

where $L_i \in Sub(L'), L_j \in Sub(L'')$. (3)

For two concepts L' and L'' , formula (3) summarizes the lexical similarity of each pair sub-concepts of them against the most sub-concepts of these two.

Based formula (2) and (3), we give the conceptual similarity formula as formula (4).

$$CSim(L', L'') = LSim(L', L'') + SupSim(L', L'') + SubSim(L', L''). \quad (4)$$

Formula (4) shows that different from the lexical measure $LSim$, the conceptual measure $CSim$ considers the lexical similarity of two concepts and the similarity of their super-concepts and sub-concepts synthetically.

3.4 Acquiring Ontological Instances

Once ontological structure is acquired and refined, the process of acquiring ontological instances, viz. step 4 in SOAM, can start. The rule is shown in rule 12.

Rule 12. For an ontological class C_i , if C_i is corresponding to relations R_1, R_2, \dots, R_i in database, then every tuple $t_i, t_i \in R_1 \bowtie R_2 \bowtie \dots \bowtie R_i$, can be mapped to a instance of C_i .

Comment: Rule 12 shows that for one ontological class, its instances consist of tuples in relations corresponding to the class and relations between instances are established using the information contained in the foreign keys in the database tuples.

4 Case Study

Now the SOAM has been applied to the Semantic Web project in Renmin University of China. There are huge resources about social science in the digital library of Renmin University of China and they are stored in relational database, so our goal is to acquire OWL ontology from it.

To prove the correctness and practicability of our method, we choose the library on economics as the experimental data and refine the obtained ontological structure based on Classified Chinese Library Thesaurus, which classifies the controlled vocabulary in some domains and is adopted widely for organizing literature resources. By using SOAM, an economics ontology has been created, which

comprises about nine hundred classes, one thousand and one hundred properties and thirty thousand instances. The classes and properties are organized in hierarchy. According to OWL specification, the non-taxonomical relations of classes are embodied by object properties. The result is shown in Fig. 2, which is the screen snapshot of the acquired economic ontology.

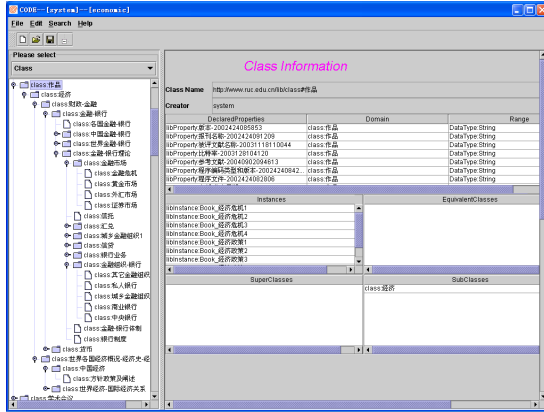


Fig. 2. Screen Snapshot of Acquired Ontology

In Fig.2, the tool named CODE[8] is developed by us for ontology construction. The left shows the hierarchy of ontological classes and the right is the corresponding properties, property restrictions and instances. The hierarchy of properties can be seen in other windows, which is not shown here. Users can modify the ontology directly in CODE according to the refinement requirement.

During the process of acquiring economics ontology, we believe that SOAM is reasonable, the related acquisition rules are effective and the refinement method can help user to validate ontology semi-automatically.

5 Related Works

Much work has been addressed on the issue of explicitly defining semantics in database schema[9,10], extracting semantics out of database schema[11] and transforming a relational model into an object-oriented model[12]. However, the semantics obtained by previous cannot meet the requirement of constructing ontology fully. Although object-oriented model is close to an ontological theory, there are still some differences between them. For example, there does not exist hierarchies and cardinality about properties in object-oriented model. So these methods cannot be used to discover ontology from relational database directly.

There are few methods investigating the transformation from a relational model to an ontological model. The method in [13] creates a new database

schema by analyzing the database schema firstly. Then the content of the new database schema is mapped into the ontology by reverse engineering techniques. During the transformation, the method uses a new database schema as middle model, which is different from ours. Moreover, the semantic characteristics of the database schema are not always analyzed. The method in [14] is proposed to automate the process of filling the instances and their attributes' values of an ontology using the data extracted from external relational sources. This method uses a declarative interface between the ontology and the data source, modeled in the ontology and implemented in XML schema. However, our method acquires ontological instances by mapping. The method in [15] is to build light ontologies from conceptual database schemas. It uses a mapping process, which is similar to our approach. However, its target is to construct RDF(S) ontology. RDF(S) has unclear semantics and has not inference model, which is incompetent for automatic tasks. Our target is to acquire OWL ontology because OWL has clear semantics brought by description logic systems, and OWL enjoys a well-founded inference model from some particular description logics. All the previous methods cannot acquire OWL ontology from relational database.

In addition, most previous methods pay no attention to the ontology refinement after acquiring ontology from relational model. However, we have discussed that ontology refinement is a necessary step during ontology construction. Although [13] proposes to refine the ontology based on user queries, there are some limits in this method:(1)The cost of refinement is large, because it refines ontology after the whole ontology is constructed. In SOAM, ontological structured is refined before acquiring ontological instances to reduce the cost of ontology modification, which has been discussed in chapter 2. (2)The quality of refinement cannot be ensured, because the refinement depends on the user queries, which is random. In our method, the ontological structured is refined according to the repositories of lexical knowledge, which is authoritative and steady.

6 Conclusion and Future Works

The paper gives a semi-automatic ontology acquisition method(SOAM) based on data in relational database and introduces the implementation of it in detail. In the end, an application case is given to prove that SOAM is practical and helpful to the automation of ontology acquisition.

In the future, we will apply our approach to acquire ontologies in other domains. And we plan to do some researches on acquiring ontology from other resources, such as natural language text, XML and so on.

Acknowledgements

The work was supported by the National Natural Science Foundation of China (Grant No. 60496325).

References

1. Berners-Lee T, Hendler J, Lassila O. The Semantic Web. Scientific American feature article. 2001
2. OWL, Web Ontology Language (OWL), <http://www.w3.org/2004/OWL/>, 2004
3. Codd EF. A relational model of data for large shared data banks. CACM 13 No.6, 1970
4. Maedche A. Ontology learning for the Semantic Web. Boston: Kluwer Academic Publishers. 2002
5. Ramanathan S, Hodges J. Reverse engineering relational schemas to Object-Oriented schemas. Technical Report No. MSU-960701. Mississippi State University. 1996
6. Salzberg B. Third normal form made easy. Sigmoid Record, 1986, 15(4):2 18
7. Man Li, Dazhi Wang, Xiaoyong Du, Shan Wang. Ontology construction for semantic web: a role-based collaborative development method. Proc. of the 7th Asia Pacific Web Conference (APWeb). 2005
8. Levenshtein, I.V. Binary codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory, 10(8):707-710
9. Rishe N. Database design: the semantic modeling approach. McGraw-Hill, 1992
10. Biskup J. Achievements of relational database schema design theory revisited. Semantics in Database, Springer Verlag, 1998
11. Chiang R, Barron T, Storey V. Reverse engineering of relational databases: extraction of an EER model from a relational database. Journal of data and knowledge engineering, 1994,12(2):107-142
12. Vermeer M, Apers P. Object-oriented views of relational databases incorporating behaviour. Proc. of DASFAA. 1995
13. Kashyap V. Design and creation of ontologies for environmental information retrieval. Proc. of the 12th Workshop on Knowledge Acquisition, Modeling and Management. 1999
14. Rubin DL, Hewett M, Oliver DE, Klein TE and Altman RB. Automatic data acquisition into ontologies from pharmacogenetics relational data sources using declarative object definitions and XML. Proc. of the Pacific Symposium on Biology. 2002
15. Stojanovic L, Stojanovic N, Volz R. Migrating data-intensive Web Sites into the Semantic Web. Proc. of the 17th ACM symposium on applied computing. 2002

Watermarking Abstract Tree-Structured Data

Gang Chen, Ke Chen, Tianlei Hu, and Jinxiang Dong

College of Computer Science, Zhejiang University, Hangzhou, P.R.China 310027
cg@zju.edu.cn, arkeke2005@yahoo.com.cn, andy_hu@263.net,
djx@zju.edu.cn

Abstract. Digital watermarking techniques have been successfully applied in multimedia data such as image, audio, video, etc. With the rapid development of Internet, the requirement of copyright and integrity protection over semi-structured data, such as XML documents and complex hypertext content, is becoming more and more urgent. However, it is more challenging to apply the effective watermarking schemes into semi-structured data, because watermark for semi-structured data must have the dual protection feature toward both structure and content. In general, most semi-structured data can be represented as tree. Therefore, in this paper, we propose a novel watermark scheme for tree-structured data based on the value lying both in the tree structure and in the node content, which gives a comprehensive protection for both node content and structure of tree. The experiments on the robustness of the solution against various forms of attacks are conducted. The results highlight correctness and efficiency of our solution.

1 Introduction

As an important area in information hiding [1,2,3], digital watermarking techniques have been extensively exploited and regarded as a potentially effective solution against illegal reproduction or theft of digital assets. With the rapid development of computer technique and the sharp increase of the capacity of storage, more and more valuable information in digital form is produced, packaged, delivered and stored expediently, which exerts tremendous pressure on data providers to allow users to search and access data remotely and to identify pirated copies of their data as well. So the ownership of digital assets has received many concerns.

There is a rich body of literature on watermarking multimedia data. Most of these techniques were initially developed for still images [4,5] and later extended to video [6] and audio sources [7]. While more recent research has been extended to some new digital domains, such as relational data [8,9], software [10,11], natural language [15,16], categorical data [17], sensor streams [18] and so on.

With the rapid development of Internet, copyright and integrity protection is becoming the most urgent requirements for semi-structured data, such as XML documents, complex hypertext content, etc. Semi-structured data can be represented as trees and graphs. Our work in this paper mainly focuses on tree-structured data. Nowadays, while the research work in watermarking semi-structured data [12,13,14] has achieved a lot, there are also new technical challenges due to tree-structured data and multimedia data differ in three important respects:

- 1) **Securing Structural Information.** Tree-structured data is composed of two parts, node content and its structural information. Its knowledge is embodied not only in the content of individual node, but also in the structure. A little changes of the structure may result in fundamental changes of the knowledge. So the watermarking technique for tree-structured data without the protection of the structural information is very fragile. Meanwhile, when dealing with trees in general, we are faced with the issue of uniquely identifying and referencing nodes, which is most vital problem of watermarking semi-structured data. While numerical value, text, image, video, or audio may be a node in tree-structured data.
- 2) **Relatively Low Bandwidth.** Most media watermarking techniques make use of the inherent large noise-bandwidth associated with Works that are to be “consumed” by the human sensory system with its limitations [12]. Thus, the watermark has a large cover in which to hide. In the case of abstract tree-structured data, watermarking bandwidth appears to be available from capacities associated to properties of both the tree structure and the data type of the composing nodes. The bandwidth of tree-structured data is relatively low for various type of the node.
- 3) **More Attacks on Structural Information.** Semi-structured data may suffer more initiative attacks on tree structural information, such as addition, deletion of nodes and relations among the nodes in the tree. In the meanwhile, the probability of the subtractive (subtree) attacks has also significantly increased.

In this paper, we introduce a novel watermarking method for tree-structured data, which embeds watermarking not only in value-carrying nodes but also in tree structure that “glue” all the nodes together. The proposed technique in the method is robust against various forms of malicious attacks not only to the tree node, but also to the tree structural information.

The rest of this paper is organized as follows: Section 2 discusses some related works. Section 3 sketches out the watermark model for tree-structured data. Section 4 then proposes a solution to watermark tree-structured data and gives our algorithms for embedding and detecting watermarks. Section 5 shows our experiment evaluation. We conclude with some closing remarks in Section 6.

2 Related Works

In recent years, the techniques about how to embed a resilient and indelible mark in semi-structured data, which can be used as copyright or other stamp information in court when these data are resold illegally by others, have received much attention and made some progresses. Agrawal et. al. [8] present an effective watermarking technique geared for relational data. His technique ensures that some bit positions of some of the attributes of some of the tuples contain specific values, which is robust against various forms of malicious attacks as well as benign updates to the data. It can be viewed as node data here, but cannot protect structural information. Ng and Lau [13] extend Agrawal’s techniques on XML data by defining locators in XML in their selective approach which allows embedding non-destructive hidden information content

over XML data. Similarly, it cannot protect structural information. They also present another different watermarking scheme: the compression approach, which takes verbosity and the need in updating XML data in real life into account. But their watermarking compression approach is only developed in their XQzip [19], which maybe limit the watermarking algorithm application greatly. Gross-Amblard et. al. [20] investigate the problem of watermarking XML databases while preserving a set of parametric queries. Sion et. al. [12] discuss the watermarking of abstract structured aggregates of multiple types of content, such as multi-type/media documents, and represents them as graphs by characterizing the values in the structure and individual nodes. He also proposes a general watermarking algorithm that makes use of the encoding capacity of different types of nodes.

In this paper, we initially draw on Agrawal's idea, randomly choose tree nodes to be watermarked by various node content, and embed the corresponding watermark in the nodes. While, the owner of these tree nodes can claim of the ownership by watermarks in these nodes, but cannot claim of the ownership of the tree structure. Therefore, the method is not so meaningful when facing the application that the tree structural information is more valuable than the tree node itself. So we develop the Path-Node Watermarking (PNW) method, which can solve the crucial problems mentioned above by use of two value facets, tree structure and tree node content.

3 The Model

The watermark model we propose here is described by some key definitions, its requirements to be met and common malicious attacks to tree-structured data.

3.1 Definitions

Definition 1. Tree-structured Watermark

Let tree-structured watermark W_T be a tuple: $W_T = \{ T, M_n, \gamma, \alpha, \tau, C, F_{ic}, K \}$ where:

- 1) T is the tree-structured data to be marked.
- 2) M_n is a set of tree-node watermarks inserted into the tree-structured data.
- 3) γ is a gap of watermarking nodes in the tree-structured data. Correspondingly, $1/\gamma$ is the watermark ratio of the watermarked data, and $\omega = N/\gamma$ is the number of marked nodes in the tree.
- 4) α is the significance level of the test for detecting a watermark.
- 5) τ is minimum number of detected nodes required for a successful detection, which is closely related to α .
- 6) C is a condition expression. Usually, when it is satisfied, the current node is selected as a candidate node for watermarking.
- 7) F_{ic} is a set of usability constraints which need to be met before and after the watermarks embedding.
- 8) K is a key used for watermarks insertion and detection, which is known only to the owner of the tree-structured data. K is used to enforce security, which prevents any unauthorized party the privilege of recovering and manipulating the watermark.

Definition 2. Tree-Node Watermark.

Tree-node watermark embodies tree-structured watermark. Let a tree-node watermark M be a quintuple $M = \{ t, d, s, \zeta, w \}$, where:

- 1) t is the type of the attribute of the node.
- 2) d is the content of the node.
- 3) S is the structural information of the node in the tree.
- 4) ζ , related to t closely, represents usability constraints needed before and after the node's watermark insertion. e.g. when the node is numeric, ζ is the number of the least significant bits available for marking in the node.
- 5) w is the watermark information embedded in the node.

For the convenience of the description in the algorithms below, table 1 states several key parameters used in our watermarking methods.

Table 1. Notation

Notation	Meaning
N	Number of nodes in the tree.
ζ	Number of the least significant bits available for marking a node.
$1/\gamma$	Fraction of marked nodes in the tree (the watermark ratio).
ω	Number of marked nodes in the tree.
α	Significance level of the test for detecting a watermark.
τ	Minimum number of detected nodes required for a successful detection.

3.2 Malicious Attacks

The tree-structured data watermark may suffer attacks as following:

A. Node-content attacks. i.e. Adversary modifies node content within usability vicinity. Generally speaking, attack manners change with the type of the node content. For example, for numeric node, a zero out attack sets values of some number of bit positions to zero, and a bit flipping attack inverts the values of some number of bit positions. For textual data, a synonym attack may replace some words in the node. etc.

B. Tree- structure transformation Attacks, mainly including:

- B1: Adversary inserts some insignificant nodes to the tree.
- B2: Adversary deletes some insignificant nodes from the tree.
- B3: Adversary changes the relation among the nodes.

C. Subtractive (Subtree) Attacks. Adversary partitions a tree into many independent usable trees, to meet the requirements of elimination of watermarks in the tree.

D. Additive Attacks. In an additive attack, adversary inserts his own watermarks over the original document and claims the "legal" ownership of the data.

4 A Solution

We now present two watermark methods for abstract tree-structured data in this section. The first adopts the traditional watermark scheme based on the node content only, while the second develops the PNW algorithm based on the value lying both in the tree structure and in the node content. The design of a mapping function to calculate the corresponding labels is at the heart of watermarking tree-structured data. It is of paramount importance that the function must have the ability to uniquely ascertain label value, and to “recognize” the labeled node after the action of watermark embedding or malicious attacking (e.g. transform tree node content or change the relations of the nodes within the usability vicinity). So, the main idea of designing the function is that we must have the label in accordance with the most valuable part (i.e. the usability of content will degrade greatly with minor changes to the part) in the tree node. In fact, the most important difference of the two schemes is how to calculate labels. The analysis of each scheme is given in the following.

4.1 Watermark Based on Tree Node Content Only

As mentioned above, inspired by the relational watermarking algorithm proposed by Agrawal [8] we bring forward a tree-structured data watermarking method based on tree node content only. The algorithm accomplishes effective protections for the value in the tree-structure’s nodes, which is robust against various attacks but short of protecting the value pertaining to the tree structure itself.

4.1.1 Labeling the Tree Nodes

Apparently, in this scheme, the node-labeling function that calculate the label for tree node content is closely related to the type of tree node content, the way of watermark embedding, and the form of envisioned attacks, which must be adaptive in our implementation.

And more specifically, if the tree node is numeric data and the watermark embedding function is to set the ζ least significant bits to 0, we can use a linear combination of the high order bits of the tree node content to computer label. For image data, if the watermark method is based on frequency domain transforms like the DCT, the label could be a combination of the most significant bits of its significant DCT coefficients. For video or audio data, feature extraction algorithms could be also explored as means to provide corresponding node-labeling functions.

4.1.2 Watermark Insertion

To better illustrate, we represent the tree structure by the aid of son-sibling method, and use *item*, *lson*, *rsibling* denoted node content, son and sibling respectively. Furthermore, we take up a one-way hash function $F(x) = H(K \cdot x)$, which returns an unique integer value depended on a input x and a private key K known only to the owner. And let *Label* () be the node-labelling function. In this algorithm, *markNode*() is self adaptive and can be extended with the real type of the node. Here we only give the mapping function *markNum*() for the numerical data due to the restriction of the paper length. The specific description of the insertion algorithm is as follows:

Algorithm 1. The watermark insertion algorithm

```

// G: the root of the tree
// K: The private key is known only to the owner of the tree-structured data
// The parameters  $r, \zeta$  are also private to the owner.
tree_wm_insertion(tree_node G, secret_key K)
1: if (G is NULL)
2:   return; // the outlet of the recursive function
3: if (G is a candidate node for marking)
4:   if ( $F(\text{Label}(G)) \bmod r$  equals 0) //mark this node
5:     markNode(G, K,  $F(\text{Label}(G))$ );
6: tree_wm_insertion (G->lson, K); //implement watermark insertion by recursion
7: tree_wm_insertion (G->rsibing, K); //implement watermark insertion by recursion
//markNode() is an adaptive design closely related to the tree node content.
markNode(tree_node T, secret_key K, sub_key sk)
8: if (T->item is numerical)
9:   markNum(T, K, sk);
10: else
11:   .....//The other types are not presented here due to the restriction of the paper
length
markNum(tree_node T, secret_key K, sub_key sk)
12:  $v = T \rightarrow \text{item}$ ;
13: bit-index  $j = H(K \cdot sk) \bmod \zeta$ ; //modify the  $j$ th bit
14: if ( $H(K \cdot sk)$  is even)
15:   set the  $j$ th least significant bit of  $v$  to 0;
16: else
17:   set the  $j$ th least significant bit of  $v$  to 1;
18:  $N \rightarrow \text{item} = v$ ;

```

4.1.3 Watermark Detection

In order to detect whether the data has originated from the original one, we have to find out the number of marked elements in the tree-structured data, denoted as *MatchCount*, and use a threshold function to calculate the smallest integer, denoted as τ , such that if the *MatchCount* is larger than τ , the owner can claim the ownership of the data with the confidence of $(1-\alpha)$. According to Agrawal's research [8], the threshold is the minimum integer k such that $\sum_{r=k}^n C_n^r p^r (1-p)^{n-r} < \alpha$. Obviously, here probability $p = 1/2$ according to the *markNum()*. The specific description of the detection algorithm is as follows:

Algorithm 2. The watermark detection algorithm

```

// G: the root of the tree
// K: The private key is known only to the owner of the tree-structured data
// The parameters  $\alpha, r$  and  $\zeta$  are also private to the owner.
tree_wm_detection (tree_node G, secret_key K)
1: totalcount = matchcount = 0;
2: get_tree_wm_count (G, K, &totalcount, &matchcount);
3:  $\tau = \text{Threshold}(\text{totalcount}, \alpha)$ ;
4: if (matchCount  $\geq \tau$ )
5:   The tree is a suspect piracy;
get_tree_wm_count (tree_node G, secret_key K, int *pTotalcount, int *pMatchcount)

```

```

6: if (G is NULL)
7:   return; // the outlet of the recursive function
8: if(G is a candidate node for marking)
9:   if (F(Label(T)) mod r equals 0) { // this node was marked
10:    *pTotalcount = *pTotalcount + 1;
11:    *pMatchcount = *pMatchcount + isMatchMark(T, K, F(Label(G)));
12:   }
    //implement watermark detection by recursion
13: get_tree_wm_count (T ->lson, K, pTotalcount, pMatchcount);
14: get_tree_wm_count (T ->rsibling, K, pTotalcount, pMatchcount);
isMatchMark (tree_node T, secret_key K, sub_key sk)
15: if (T->item is numerical) {
16:   v = T->item;
17:   bit_index j = H(K · sk) mod ζ; // the jth bit was modified
18:   if (H(K · sk) is even)
19:     return 1 if the jth least significant bit of v is 0 else return 0;
20:   else
21:     return 1 if the jth least significant bit of v is 1 else return 0;
22: } //The other types are not presented here due to the restriction of the paper length

```

4.1.4 Analysis

According to Agrawal's experiments [8], the method is robust against various forms of malicious attacks to the tree-structured data. However, if nodes content in whatever tree structure are same, this method cannot protect the tree structure efficiently. Isomer, for instance, is one of two or more substances with identical molecular formulas but different configurations, differing only in the arrangement of their component atoms. The two classes of hydrocarbon, shown in figure 1 and figure 2, are isomers. Obviously, if a chemist discovers a new structure with the same atoms of constituent elements, he cannot protect his discovery effectively by the watermark method mentioned in the section, even though watermark is not the most effectual means to patent protection.

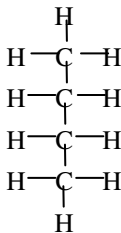


Fig. 1. The structure of butane (C_4H_{10})

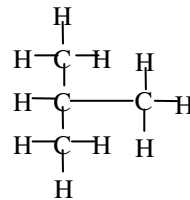


Fig. 2. The structure of isobutene (C_4H_{10})

4.2 Path-Node Watermark

In this section, we novelly propose a watermark method for abstract tree-structured data based on the technique mentioned above, which achieves comprehensive protection for tree node content and structural info as well and overcomes the defects of the former method. Before the description of this method, we enunciate one basic property of tree, which is the theory foundation of the solution in the section.

Property 1. All the nodes traversed from the node V tracing back to the root R of the tree determines identically the structural information of the tree node V .

Naturally, the watermark insertion algorithm and detection algorithm are given in the following.

4.2.1 Watermark Insertion

Compared with Algorithm 1, the nodes chosen to watermark insertion are not only based on the node content, but also on the path information (nodes traversed from the node tracing back to the root of the tree), for which the scheme is called Path-Node Watermarking (PNW). The array $a[]$ stands for the “weight” of the node content and path information respectively in the data, which are control and adjust by the user against various attacks. The $markNode()$ here is the same as which in Algorithm 1.

Algorithm 3. The PNW insertion algorithm based on combination information

```

// G: the root of the tree
// K: The private key is known only to the owner of the tree-structured data
// The parameter  $r, \zeta, a[]$  are also private to the owner.
tree_wm_insertion (tree_node  $G$ , secret_key  $K$ )
1: if ( $G$  is NULL)
2:   return; // the outlet of the recursive function
3: if ( $G$  is a candidate node for marking){
4:   tree_node  $T = G$ ;
5:   int  $L = 0, i = 0$ ;
6:   while ( $T$  is not NULL){//calculate the combination value of structural and node content information
7:      $L = L + a[i]*Label(T)$ ;
8:      $T = Parent(T)$ ;
9:      $i = i + 1$ ;
10:  }// while
11:  if ( $F(L) \bmod r$  equals 0) // mark this node
12:    markNode( $G, K, F(L)$ );
13: } // if  $G$  is a candidate node for marking
13: tree_wm_insertion ( $G \rightarrow lson, K$ ); //implement watermark insertion by recursion
14: tree_wm_insertion ( $G \rightarrow rsibling, K$ ); //implement watermark insertion by recursion

```

4.2.2 Watermark Detection

The Watermark detection algorithm is shown as follows, and the $isMatchMark()$ here is the same as that in Algorithm 2:

Algorithm 4. The PNW detection algorithm based on combination information

```

// G: the root of the tree
// K: The private key is known only to the owner of the tree-structured data
// The parameter  $a, r, \zeta, a[]$  are also private to the owner.
tree_wm_detection (tree_node  $G$ , secret_key  $K$ )
1: totalcount = matchcount = 0;
2: get_tree_wm_count ( $G, K, \&totalcount, \&matchcount$ );
3:  $\tau = Threshold(totalcount, a)$ ;
4: if ( $matchCount \geq \tau$ )

```

```

5: The tree is a suspect piracy;
6: get_tree_wm_count (tree_node G, secret_key K, int *pTotalcount, int *pMatchcount)
7:   if (G is NULL)
8:     return; // the outlet of the recursive function
9:   tree_node T = G;
10:  int L = 0, i = 0 ;
11:  while (T is not NULL){
12:    L = L + a[i]*Label(T);
13:    T = Parent(T);
14:    i = i + 1;
15:  }// while
16:  if (G is a candidate node for marking)
17:    if (F(L) mod r equals 0) { // this node was marked
18:      *pTotalcount = *pTotalcount + 1;
19:      *pMatchcount = *pMatchcount + isMatchMark(T, K, F(L));
20:    }
    //implement watermark detection by recursion
21:  get_tree_wm_count (G ->lson, K, pTotalcount, pMatchcount);
22:  get_tree_wm_count (G ->rsibling, K, pTotalcount, pMatchcount);

```

4.2.3 Analysis

Figure 3 shows the proportion of correctly marked nodes required for a successful detection with 99% confidence against different watermark ratios. We can notice that the required proportion of correctly marked nodes decreases as the percentage of marked nodes increases, and also as the watermark ratio $1/\gamma$ in the relation increases. Similarly, we find the proportion is amazingly low in order to attain 99% confidence. For example, in a tree with 100,000 nodes, if 1% of the nodes are marked, only 62% of correctly detected marks are needed to provide 99% confidence. With the tree nodes and the watermark ratio increasing, the percentage can be made less than 51%, approaching to 50%. Definitely, the proportion must be more than 50%, to differentiate a watermark from a chance of random occurrence.

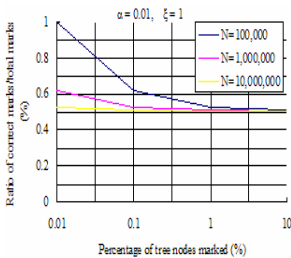


Fig. 3. Proportion of correctly marked nodes needed for detectability

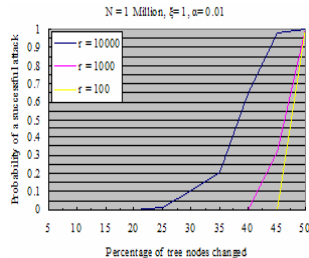


Fig. 4. Probability of a successful attack

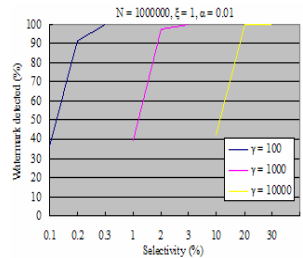


Fig. 5. Percentage of watermark detected

From the analysis above, we can draw an important conclusion that: the aggregative tree structure will “amplify” the resilience ability of the individual nodes; on the global view, the tree-structured data watermark method can maintain the detectability of the watermark, even if the individual nodes are fragile against many attacks.

5 Experiments and Resilience Analysis

We developed and implemented the algorithms on a machine of the configuration as follows: CPU: PIV 2.4G, main memory: 512M, hard disk: 40G.

For the convenience of experiments, we develop a Tree Random Generator (TRG) which can generate tree-structured data randomly. The nodes in the tree are all numeric ones for simplify the experiments.

We now report a series of experimental results for all kinds of attacks mentioned above:

Node-Content Attacks. We take $\alpha = 0.01$, $\zeta = 1$, a random tree-structured data with N set to 1,000,000 generated by the TRG, insert watermarks by various value of γ , then choose tree nodes randomly for bit flipping attack, and finally count the probability of success of the above attack on the 1 million nodes (The watermark confidence under 99% means attack success). Figure 4 shows the probability of attack success decreases quickly as value of γ declines. For $\gamma = 10,000$, there is about 63% chance of destroying the watermark with modifying randomly 40% of the tree nodes. For $\gamma = 1000$, there is only about 30% chance of destroying the watermark with modifying randomly 45% of the tree nodes. Still, the probability to destroy the watermark for $\gamma = 100$ is quite low. And when $\gamma = 10$, the watermarks in the tree-structured data are almost indestructible against node-content attacks. The results in the figure 4 illustrate that the PNW algorithm has more resilience to the attacks on node content.

Tree-Structure Transformation Attacks. The ability against attacks on tree structure transformation is distinctive of the watermarks on the abstract tree-structured data. Immunity from attacks on tree structure transformation is the principal contribution of the PNW algorithm:

For *attacks on addition of nodes* to the tree, for example, we randomly select a node as father node and insert a relatively insignificant node. Obviously, the addition does not influence any node content, and also not alter the path info from the original node traced to the root of the tree, which consequently does not affect the search of the watermarked nodes. Theoretically speaking, the PNW algorithm can identify the watermarked nodes accurately in addition attacks.

For *attacks on deletion of nodes* to the tree, for example, we randomly select a leaf node to delete. The deletion also does not influence any node content, not alter the path info, and not affect identification of the rest watermarked nodes. The only way of successful attacks may be the deletion of too much nodes, which may cause the watermarked nodes excessive loss. We cannot detect the watermarks correctly if the number of the deleted nodes is up to a threshold, so that attacks on deletion nodes degrade to subtractive (subtree) attacks in this condition.

We eliminate tree nodes randomly to calculate the probability of success attack on the 1 million nodes with the former parameters remaining unchanged. In figure 5, we have plotted the proportion of watermark detected for various γ . As expected, when γ increases (i.e. less nodes are marked), we need a lower percentage for deletion to be able to detect the watermark. For $\gamma = 100$, the watermarks are detected even if only about 0.3% subtrees remaining. And even for $\gamma = 10,000$, the attacks for watermarks are failed in need of about 20% reserved subtrees. So the PNW algorithm is also resistant to the attacks on tree node deletion.

Subtractive (Subtree) Attacks. Subtractive attacks mean that partition a tree into many independent usable trees. In order to survive these attacks, the watermarks have to be preserved also in every tree partition. In the main, there are two methods to divide a tree into many subtrees. The first is that the divided subtrees must include the original tree root, in most instances, without which the subtree may lose much valuable semantic information and even lead to the subtree useless. This case is equivalent to the node-deletion attack and the specific analysis is given above. The second is that the root of the subtree is a common node in the original branches. To resist this kind of attacks, the owner of the tree data can set the value of items with bigger index in the array a (used in Algorithm 3) to 0, according to the usability domains of all possible tree partitions. The next analysis is a repetition to the above basically. So we can conclude that the PNW algorithm is also resilience to the subtractive attacks.

Additive Attacks. In the additive attack, adversary simply inserts his own watermark in the watermarked data and claim ownership. The original ownership claim can be resolved by locating the overlapping regions of the two watermarks in which the bit values of the marks conflict and determining which owner's marks win. The winner must have overwritten the loser's bits and hence must have inserted the watermark later. Depending upon the significance level α chosen for the test, it is possible not to reach a decision if only a few marks collide. Clearly, adversaries may try to reduce the overlapping regions by using a low watermark ratio such as 0.1% or 0.01%. Similarly, the true owner of the data can confront the additive attacks by decreasing γ to increase the density of the watermarks.

6 Conclusion

The main contributions in this paper can be reduced to the following: 1) give the formalized definitions of the abstract tree-structured watermark, which is an important part of watermarking semi-structured data; 2) analyze the various forms of possible malicious attacks to tree-structured data, especially those to tree structure; 3) propose a new watermark scheme for tree-structured data based on the combination of node content as well as path information, which gives a comprehensive protection for both node content and structure of tree; 4) accomplish the experiments on tree-structured data generated randomly. From the results, the PNW solution shows greater resilience to the malicious attacks mentioned above.

In the future, we would like to extend the PNW technique to also mark non-tree structure, such as graph, and also focus on marking various types of nodes.

References

1. Ross J. Anderson (ed.). Information Hiding, First International Workshop. Lecture Notes in Computer Science, Vol. 1174. Springer-Verlag, Cambridge, U.K. (1996)
2. Fabien A. P. Petitcolas, Ross J. Anderson and Markus G. Kuhn. Information Hiding-A Survey. In Pro: the IEEE, special issue on protection of multimedia content, Vol.87(7), (1999)1062-1078
3. W.Bender, D.Gruhl, N.Morimoto, and A.Lu. Techniques for Data Hiding, IBM Systems Journal, Vol.35(3/4), (1996)313-336

4. Tirkel, G. Rankin, R. van Schyndel, and et al. Electronic water mark. In: Proceedings of Digital Image Computing - Techniques and Applications (DICTA 1993) 666-672
5. J.J.K. Ó Ruanaidh, W.J. Dowling, F.M. Boland. Watermarking digital images for copyright protection. I.E.E Proceeding on Vision, Signal and Image Processing, Vol.143 (4), (1996)250-256
6. Frank Hartung and Bernd Girod. Watermarking of uncompressed and compressed video. Signal Processing, Vol.66 (3), (1998)283-301
7. Laurence Boney and Ahmed H. Tewfik and Khaled N. Hamdy. Digital watermarks for audio signals. In Proc. IEEE international conference on multimedia computing and systems, Hiroshima, (1996)473-480
8. Rakesh Agrawal, Peter J. Haas, Jerry Kiernan: Watermarking relational data: framework, algorithms and analysis. VLDB J. 12(2), (2003)157-169
9. R. Sion, M. Atallah, and S. Prabhakar. Rights protection for relational data. IEEE Transactions On Knowledge And Data Engineering, 16(6), (2004)1-17.
10. J. Nagra, and C. Thomborson. Threading software watermarks. In: Proceedings of Sixth International Workshop on Information Hiding (IH 2004). 208-223
11. Lin Yuan, P.R. Pari, and Gang Qu. Soft IP protection: watermarking HDL codes. In: Proceedings of Sixth International Workshop on Information Hiding (IH 2004). 224-238
12. R. Sion, M. Atallah, and S. Prabhakar. Resilient information hiding for abstract semi-structures. In: Proceedings of Second International Workshop on Digital Watermarking (IWDW 2003)141-153
13. Wilfred Ng and Lau Ho Lam. Effective Approaches for Watermarking XML Data. DASFAA 2005, LNCS 3453, (2005)68-80
14. S. Inoue et al. A Proposal on Information Hiding Methods using XML. In the First NLP and XML Workshop, Tokyo, Japan, (2001)
15. M.J. Atallah, V. Raskin, C.F. Hempelmann, and et al. Natural language watermarking and tamperproofing. In: Proceedings of Fifth International Workshop on Information Hiding (IH 2002)196-212
16. Yuei-Lin Chiang, Lu-Ping Chang, Wen-Tai Hsieh, and et al. Natural language watermarking using semantic substitution for Chinese text. In: Ton Kalker, Ingemar J. Cox, Yong Man Ro, Digital (eds.): Watermarking: Second International Workshop, IWDW 2003. Lecture Notes in Computer Science. Springer-Verlag, Seoul, Korea (2003) 129-140
17. R. Sion. Proving ownership over categorical data. In: Proceedings of the IEEE International Conference on Data Engineering (ICDE 2004)584-596
18. R. Sion, M. Atallah, S. Prabhakar. Resilient rights protection for sensor streams. In: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB 2004)732-743
19. J. Cheng and W. Ng. XQzip: Querying Compressed XML Using Structural Indexing. In Proc. of the EDBT, (2004)219-236
20. D. Gross-Amblard. Query-preserving watermarking of relational databases and XML documents. In: Proceedings of the 22nd Symposium on Principles of Database Systems (PODS 2003)191-201

Location-Based Caching Scheme for Mobile Clients

KwangJin Park, MoonBae Song, and Chong-Sun Hwang

Dept. of Computer Science and Engineering, Korea University,
5-1, Anam-dong, Seongbuk-Ku, Seoul 136-701, Korea
{kjpark, mbsong, hwang}@disys.korea.ac.kr

Abstract. The Voronoi Diagram(VD) is the most suitable mechanism to find the nearest neighbor(NN) for mobile clients. In NN query processing, it is important to reduce the query response time, since a late query response may contain out-of-date information. In this paper, we study the issue of location dependent information services(LDISs) using a VD. To begin our study, we first introduce a broadcast-based spatial query processing methods designed to support NN query processing. In further sections, we introduce a generic method for location-dependent sequential prefetching and caching. The performance of this scheme is studied in different simulated environments. The core contribution of this research, resides in our analytical proof and experimental results.

Keywords: Air index, wireless data broadcasting, mobile computing.

1 Introduction

Over the last two decades, several studies have been conducted with emphasis on spatial database processing. Location-dependent information services(LDISs) is one of the applications which is gaining increasing attention. LDISs represent new innovating ways to satisfy customer needs, such as traffic reports, hotel information and weather broadcasts [10]. A core concept of LDISs is the NN query, which retrieves the data closest to a query point.

In broadcast environments, the server continuously sends data to the clients, without the any specific client having to send an actual request. Any number of clients can monitor and retrieve data from the broadcast. Therefore, the client is responsible for filtering specific desirable data. If the data is efficiently organized to represent the needs of the clients, such a scheme makes effective use of low wireless bandwidth and is ideal for achieving maximum scalability. Two key requirements for data access in wireless environments are power conservation and the minimization of client waiting time. In push-based systems, the mobile clients must wait until the server broadcasts the desired data. In this case, the client waiting time is determined by the overall usage of the broadcast data channel [1]. A technique used to address this issue, called **air indexing**, operates by interleaving indexing information among the broadcast data items. At the same time, the client device can reduce its battery power consumption through the use

of selective tuning [2,4]. The Air Indexing technique can be evaluated in terms of the following factors:

- *Access Time*(AT): The average time elapsed from the moment a client issues a query to the moment when the required data item is received by the client.
- *Tuning Time*(TT): The amount of time spent by a client listening to the channel.

The *Access Time*: consists of two separate components, namely:

- *Probe Wait*: The average duration for getting to the next index segment. If we assume that the distance between two consecutive index segment is L , then the probe wait is $L/2$.
- *Bcast Wait*: The average duration from the moment the index segment is encountered to the moment when the required data item is downloaded.

The *AT* is the sum of the *Probe Wait* and *Bcast Wait*, and these two factors work against each other [2,4].

In this paper, we attempt to reduce both the TT and AT in the wireless mobile computing environment. After pointing out the limitations of the existing indexing schemes, we present various schemes that can overcome these problems. We believe this is the first work in which NN query processing without an index segment is proposed, providing the optimum access time. Throughout this paper, we assume that the data objects are in 2-dimensional space and are static, such as restaurants, hospitals, and hotels. The mobile clients can identify their locations using systems such as the Global Positioning System(GPS).

2 Related Work

Spatial databases have been studied extensively during the last two decades, from this research, several spatial access methods have been proposed. In this section, we provide some background on the location model, caching models and air index schemes, which we adapt in this study.

2.1 Data on Air

The broadcasting of spatial data together with an index structure is an effective way of disseminating data in a wireless mobile environment. This method allows mobile clients requesting data to tune into a continuous broadcast channel only when spatial data of interest and relevance is available on the channel, thus minimizing their power consumption [2,3,8].

2.2 Voronoi Diagrams

VDs are fundamental tools used to express the proximity of geometric objects. The VD divides a space into disjoint polygons where the nearest neighbor of any

point inside a polygon is the generator of the polygon. The VD for n objects on a plane can be constructed at $O(n \log n)$ complexity using a simple sweep algorithm. The general definition of the Voronoi Cell(VC) of a point in the d -dimensional space \mathbb{R}^d follows:

Theorem 2.2.1: If n is a d -dimensional point, N is a set of n points in the d -dimensional space \mathbb{R}^d , then $\mathcal{V}(p)$, the VC of the point n given set N , is defined as the unique convex polygon which contains all the points in the set $\mathcal{V}(p)$:

$$\mathcal{V}(pi) = \{q \in \mathbb{R}^d \mid \forall pi \in N, dist(q, pi) < dist(q, pj)\}, \text{ where } j \neq i. \quad (1)$$

3 VD-Based Indexing for NN Search

In this section, we first introduce the broadcast-based LDIS scheme(BBS) [8]. Then, we describe the VD-based Energy Efficient Selective Tuning method. Finally we present adjacency data prefetching and caching methods in a way with the aim of reducing the client's access time and energy consumption.

3.1 Broadcast Sequence for NN Search

In a recent paper [8], we have proposed the concept of data sorting for broadcasting called BBS(Broadcast-based Location Dependent Data Delivery Scheme). In the BBS method, the server periodically broadcasts the IDs and coordinates of the data objects, without an index segment, to the clients, and these broadcasted data objects are sorted sequentially, according to the location of the data objects, before being sent to the clients. The BBS provides the fastest access time, since no index is broadcasted along with the data and thus, the size of the entire broadcast cycle is minimized. A preliminary simulation-based results showed that BBS is significantly reduce the AT.

A simple sequential broadcast can be generated by linearizing the two dimensional coordinates in two different ways: i.e. horizontal broadcasting(HB) or vertical broadcasting(VB). In HB, the server broadcasts the LDD(location dependent data) in horizontal order, that is, from the leftmost coordinate to the rightmost coordinate. On the other hand, in VB, the server broadcasts the LDD in vertical order, that is, from the bottom coordinate to the top coordinate. In this paper, we assume that the server broadcasts the data objects using HB.

Definitions of Symbols and Parameters:

- S : Server data set
- T_i : boundary lines of the current broadcast data object
- T : set of T_i
- O_i : broadcast data object, where $O_i \in S$
- O_{candi} : candidate for the nearest data object

- O_c : current broadcast data object (initially O_c regarded as NN), where $O_c \in S$
- O_{ps} : one of the data items broadcast before O_c in the current broadcast cycle, where $O_{ps} \in S$
- O_p : a data item broadcast just before O_c in the current broadcast cycle, where $O_p \in S$
- O_{FP} : data object of FP
- O_f : the client's first tuned data item in the broadcast channel
- Data_first: the server's first broadcast data item in the current broadcast cycle.
- TN : nearest boundary line on the left-hand side of the q , e.g., T5 in Figure 1
- ON : data object of TN , e.g., O16 in Figure 1
- TS : safe nearest boundary line on the left-hand side of the q , where x-coordinate of $TS \leq$ x-coordinate of TN , e.g., T4 in Figure 1
- TS_{candi} : candidate of TS

3.2 VD-Based Energy Efficient Selective Tuning

BBS scheme[8] can significantly reduce the average access time, since it eliminate the probe wait time for the clients. However, tuning time may increase, since the client has to tune the broadcast channel until the desire data item is arrived.

In this section, we present an energy efficient scheme under the BBS environment namely, VD-based Selective Tuning(VST) method. The VCs can be used to find out NN. The present scheme provides the ability to selective tune the clients and therefore helps reduce the client's tuning time.

VD-Based Selective Tuning(VST). In this section, we present a selective tuning method for use in the BBS environment. In this method, the client uses exponential pointers from each data item for the purpose of reducing energy consumption. Each data item contains the following information:

- It's ID and VE(Voronoi Edge: vertex pointers)
- Initial Pointer: arrival time of the first broadcast data item for the next broadcast cycle.
- Forward Pointer(FP): IDs, it's VE and arrival times of the data items that will be broadcasted at T_i . $T = \sum_{i=1}^{\log_e N} T_i$, from the each data item, where N be the number of data items and e be exponent value(e.g., if $e=2$, the data item 1 has the following IDs and related information: the data items located in T2, T4, T8, T16, and T32(see Figure 1)).

The client obtains the ID, VE, location information and FP from the initial data obtained at the broadcast channel. Then, it checks the VE of the current data item and FP in order to find out the object that contains query point q within the VC. If there is no data object that contains query point q within the

cell, then it switches to doze mode until the desired data item appears on the broadcast channel. The client repeatedly switches between the doze and wake up modes until it retrieves the NN.

Let us consider the example in Figure 1. First, the client tunes into the broadcast channel at T_1 and obtains the following FP: $ID=\{2,4,8,16,32\}$, their arrival times and VEs from data item O_1 . Then the client checks the VEs of each date object $O_1, O_2, O_4, O_8, O_{16}$ and O_{32} in order to find out the VC that contains query point q . The client switches to doze mode, since VC of data object $O_1, O_2, O_4, O_8, O_{16}$ and O_{32} does not contains query point q . The client sleeps until O_{16} (at T_5) has appeared on the broadcast channel, since T_5 is the nearest boundary line on the left-hand side of the query point q up to the present time. Then, the client wakes up at T_5 , and obtains FP $ID=\{17,19,23,31\}$ from data item O_{16} , and checks the VC of each date object O_{17}, O_{19}, O_{23} and O_{31} . The client switches to doze mode, since VC of data object O_{17}, O_{19}, O_{23} and O_{31} does not contains query point q . Thus, the client again switches to doze mode until O_{23} (at T_4') has appeared on the broadcast channel, since T_4' is the nearest boundary line on the left-hand side of the query point q up to the present time. Then, the client wakes up at T_4' and obtains FP $ID=24, 26, 30$ from data item O_{23} , and checks the VC of each date object O_{24}, O_{26} and O_{30} . Finally, the client returns data item O_{26} as the NN, since VC of the data object O_{26} contains the query point q .

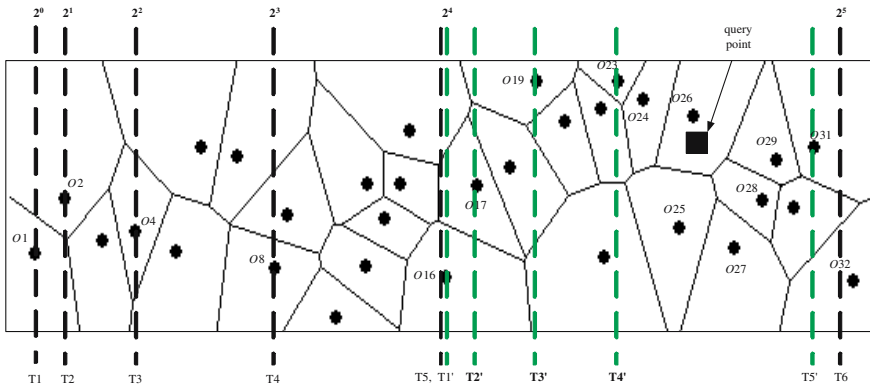


Fig. 1. An example of Exponential Sequence Scheme

Theorem 3.2.1: While the data objects are sequentially broadcasted in horizontal order, from the leftmost coordinate to the rightmost coordinate, if x -coordinate of $O_f > x$ -coordinate of q , then the client cannot guarantee that it dose not miss NN in the current broadcast cycle.

The client uses following algorithm to find out NN:

Algorithm 1. the client algorithm used to identify the nearest object

Input: locations of the clients and the data objects;
Output: NN;
Procedure:

```

1: do {
2:   read  $O_i$ 
3:   if(x-coordinate of  $O_f >$  x-coordinate of  $q$  and  $O_f \neq$  Data_first)
4:     then switch to doze mode until Data_first comes
5:      $O_i =$  Data_first
6:   else if (satisfy the Lemma 3.2.1)
7:     then switch to doze mode until Data_first comes
8:      $O_i =$  Data_first
9:   else
10:    for  $T$ 
11:      do check VE of  $O_i$  and  $O_{FP}$  to identify that VC of  $O_i$  or  $O_{FP}$  contains query point  $q$ 
12:      if VC of  $O_i$  or  $O_{FP}$  contains query point  $q$ 
13:        then return  $O_i$  or  $O_{FP}$  as NN
14:      else
15:        then find  $TN$  on the left hand side of  $q$ 
16:        and  $TS$ , then switch to doze mode until the
17:        data object of  $TS$  appears on the channel
18:         $O_i =$  object of  $TN$ 
19:    }
20: } while(find out NN)

```

Theorem 3.2.2: To guarantee that the client does not miss the NN, the $\text{dist}(\text{x-coordinate of } TS, \text{x-coordinate of } q)$ must be longer than $\text{dist}(ON, q)$.

3.3 Adjacency Data Caching(ADC)

In this method, the client prefetches the data objects that are adjacently located in the map for future use.

Given a VD and the query point q , we can identify the VC containing q . Moreover, since the broadcasted data objects are sorted sequentially according to the location of the data objects, clients can prefetch adjacent located data objects. Let w_p denote the size of prefetched data objects. Prefetching can be categorized into pre-fetch and post-fetch, where pre-fetch represents fetching data objects before NN data item is broadcast while post-fetch represents fetching data objects after the NN data object is broadcasted respectively. The value of pre-fetch and post-fetch represents the number of data items to be prefetched. The client adjusts the size of w_p according to the client's moving direction, speed or the cache size. It should be noted that prefetching does not necessarily require continuous active monitoring of the broadcast channel to determine what the next arriving data object is, if the client knows in advance the broadcast schedule. Cached data items contain the information of the object's ID and it's VE. Let us consider the case where the server broadcasts data items using the BBS and the client detects the nearest data object from the query point q as

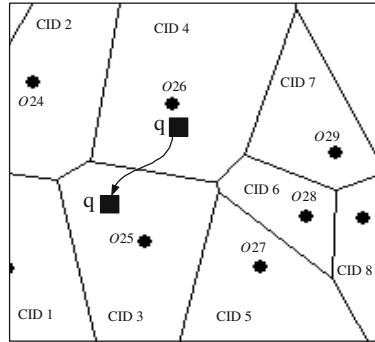


Fig. 2. Broadcast sequence= {O1., O24, O25, O26, O27, O28, O29., On} and Cache state={O24, O25(NN, when the client located in CID3), O26(NN, when the client located in CID4), O27, O28, O29}

O26. Let the value of the pre-fetch be 2 and post-fetch be 3 on the basis of the NN. In this case, the client wakes up at T4' and prefetch data objects as the following sequential order:={O24, O25, O26, O27, O28, O29}, as shown in Figure 2. The client cached the data object from O24 to O29 while it located at Voronoi Cell ID(CID) 4. Then, the client moves from CID 4 to CID 3(see Figure 2). The client checks the prefetched data items in CID 3 instead of tuning the broadcast channel again and returns O25 as newly nearest data object, if it receives the same query.

Definition 3.3.1: x -coordinate of pre-fetch \leq x -coordinate of NN \leq x -coordinate of post-fetch.

Although the previous index schemes, such as (1,m) index can also be adapted in ADC, in this case, AT and TT may significantly increase, since the client has to repeatedly switch between the doze and wake up modes until it obtains the desired data items.

Let $V(O_x^c)$ be the validity region of data item in the cache and S^c be the set of data items in the cache. The following summarizes the steps taken by the client to process the NN search:

Step 1: When a query is issued by a mobile client, it first checks the $V(O_x^c)$, where $V(O_x^c) \in S^c$. If $V(O_x^c)$ contains the query point q, go to step 2; otherwise, go to step 3.

Step 2: If $V(O_x^c)$ contains the query point q, the data item is retrieved locally. Go to step 5.

Step 3: If $V(O_x^c)$ does not contain the query point q, the client starts tune the broadcast channel in order to process the NN query.

Step 4: The client switches to doze and wake up mode until NN has appeared on the broadcast channel. Obtain the NN through the broadcast channel.

Step 5: A result is returned to the client.

4 Performance Evaluations

In this section, we evaluate the performance with various kinds of parameters settings such as the client's speed, the size of the service area, and the distributions of the data items. In section 4.1, we analyze the proposed approaches. Then, we present performance results of simulation experiments in section 4.2.

4.1 Analytic Evaluation

Access Time. In this section, we compare the access time of the BBS with (1,m) index. The following shows comparison of the *Probe Wait* and the *Bcast Wait* between BBS and previous index method [4,2]. Let *AAT* be the average access time, *m* be the number of times broadcast indices, *N* be the number of data items and *C* be the average number of buckets containing records with the same attribute value:

Probe Wait:

- Previous index method: $\frac{1}{2} * (index + \frac{N}{m})$
- BBS method: *None*

Bcast Wait:

- Previous index method: $\frac{1}{2} * ((m * index) + N) + C$
- BBS method: $\frac{1}{2} * N + C$

Since the *AT* is the sum of the *Probe Wait* and the *Bcast Wait*, average *AT* for:

Previous Index Method Is:

$$\begin{aligned} AAT_{PRE} &= \frac{1}{2} * (index + \frac{N}{m}) + \frac{1}{2} * ((m * index) + N) + C \\ &= \frac{1}{2} * ((m + 1) * index + (\frac{1}{m} + 1) * N) + C. \end{aligned} \quad (2)$$

BBS is:

$$AAT_{BBS} = \frac{1}{2} * N + C. \quad (3)$$

Tuning Time. In this section, we evaluate the tuning time for the proposed schemes with (1, m) index. The probability distribution of the initial probe of clients is assumed to be uniform within a broadcast and data items of the same size.

Let *ATT* be the average tuning time and *l* be the number of levels in the multileveled index tree. The *ATT* for (1,m) index is:

$$ATT_{PRE} = 1 + l + C \quad (4)$$

The *ATT* of BBS is as follows:

Let m denote the number of times broadcast indices, k' denote the number of levels in the index tree for BBS and P denote the probability:

P {containing the desired data object among the index} is $\frac{1}{m}$, and then, P {obtaining the desired data object} is $\frac{1}{2m}$.

Thus, *ATT* of BBS is P {obtaining data object without an index} \times cost of reading data objects + P {failure obtaining the desired data object after read the index} \times cost of obtain the desire data object after read the index, and thus,

$$\begin{aligned} f(m) &= \frac{1}{2m} \left(\frac{Data}{m} \times \frac{1}{2} \right) + \frac{2m-1}{2m} \left(\frac{Data}{m} \times \frac{1}{2} + k' + 1 \right) \\ &= \frac{Data - k' - 1}{2} m^{-1} + k' + 1, \text{ thus} \\ ATT_{BBS} &= \frac{Data - k' - 1}{2} m^{-1} + k' + 1 \end{aligned} \quad (5)$$

Finally, we evaluate the *ATT* for VST. Let N be the number of data items, e be the exponent value and ATT_{VST} be the average tuning time for VST. The minimum number of steps is 1 and the maximum number of steps is $k-1$, where $k = \lceil \log_e N \rceil$. For example, if $N=1024$ and $e=2$, then in the best case, the client obtains the desired data item within a single step while, in the worst case, the client obtains the desired data item within 9 steps. The frequency of the worst case for N is 1, while the frequency of the best case for N is k .

The *ATT* for:

$$\begin{aligned} ATT_{VST} &\approx \frac{k \times 2^{k-1} \times \sum_{i=0}^{e-1} i}{N} \\ &\approx \frac{k \times e(e-1)}{4} \end{aligned} \quad (6)$$

4.2 Experimental Evaluation

This section presents the numerical results obtained through analysis and simulation. For fairness, we use a single dedicated machine. The machine is a PC with a Pentium III 800 MHz, 512 MB, and running on Windows 2000. We implemented the system in Java and run it under the JVM version 1.3.0. We evaluated two different parameters: energy consumption and access time. We assume that the broadcast data objects are static, such as restaurants, hospitals, and hotels. We use a system model similar to that described in [5,6]. The whole geometric service area is divided into groups of MSSs (Mobile Supporting Stations). In this paper, two datasets are used in the experiments. The first data set $\mathcal{D}1$ contains data objects randomly distributed in a square Euclidian space, while the second data set $\mathcal{D}2$ contains the data objects of hospitals in the Southern California area, and is extracted from the data set at [7]. In this experiment, we assume that the client's mobility pattern follows Random Waypoint Mobility Model [12].

Access Time. In this section, we evaluate the Access Latency for various parameter settings. Figure 3(a),3(b) and 3(c) show the results of the access latency as the number. of clients, size of data and the frequency of data increases in \mathcal{D}_1 respectively.

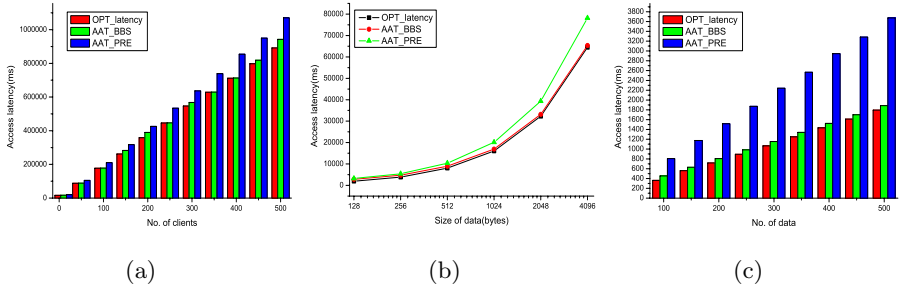


Fig. 3. Access latency in \mathcal{D}_1

Finally, Figure 4(a) and 4(b) show the result of the access latency as the number of clients and the size of data increases in \mathcal{D}_2 respectively.

Cache Hits. This section evaluates the cache hit ratio for various parameters settings such as the size of the cache, the client’s speed and the number of clients. First, we vary the size of the cache from 3072 to 11264 bytes. In this experiment, we assume that all data items are of the same size, for example 1024bytes. Figure 5(a) shows the result of the number of cache hits as the size of the cache is increased. As shown in the figure, ADC(Adjacency Data Caching) outperforms (1,m) index method, since the ADC caches the sequentially ordered data items according to their locations, whereas the (1, m) index method caches the irregularly ordered data items. Evidently, in the (1, m) index method, the average access time may significantly increase if the client is attempting to cache the data items with the sequential order, since the broadcast data items are not

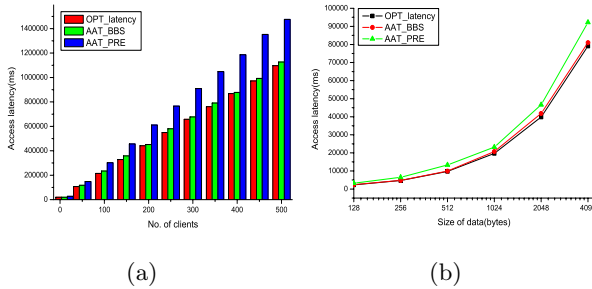


Fig. 4. Access latency in \mathcal{D}_2

ordered based on their locations. Second, we vary the client's speed from 10 to 100 in $\mathcal{D}1$. As shown in Figure 5(b), the number of cache hits decreases as the client's speed increases. Third, we vary the number of clients from 10 to 50 in $\mathcal{D}1$. As shown in Figures 5(b) and 5(c), ADC outperforms the (1,m) index method for the same reason shown above.

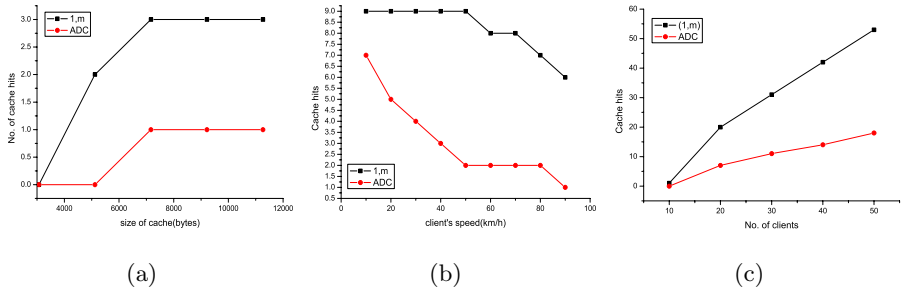


Fig. 5. Number of cache hits

5 Conclusion

We have explored the different broadcasting and tuning methods, which can be used for NN query processing. For the purpose of broadcasting in LDISs, we present the BBS and for the purpose of selective tuning with the BBS, we present the VST method. The BBS method attempts to reduce the Access Latency and the VST method attempts to conserve battery power consumption. With the proposed schemes, the client can perform NN query processing without having to tune an index segment. We also present the proposed schemes as investigated in relation to various environmental variables. The experimental results show that the proposed BBS scheme significantly reduces access latency, since the client does not always have to wait for the index segment. The resulting latency and tuning time is close to the optimum as our analyses and simulation results indicate. In a future study, we plan to investigate the cache replacement scheme.

References

1. X. Yang and A. Bouguettaya, "Broadcast-Based Data Access in Wireless Environments," In *Proc. of EBDT*, 2002.
2. T. Imielinski, S. Viswanathan, and B.R.Badrinath, "Energy efficient indexing on air," In *Proc. of SIGMOD*, 1994.
3. Y. D. Chung and M. H. Kim, "An Index Replication Scheme for Wireless Data Broadcasting," *Journal of Systems and Software*, 2000.
4. T. Imielinski, S. Viswanathan, and B.R.Badrinath, "Data on Air: Organization and Access," *IEEE Trans. Knowledge and Data Eng*, 1997.
5. B. Zheng, J. Xu, and D. L. Lee, "Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments," *IEEE Trans. Comp.*, 51(10), 2002.

6. D. Barbara, "Sleepers and Workaholics: Cashing Strategies in Mobile Environments," In *Proc. of SIGMOD*, 1994.
7. Spatial Datasets, <http://dias.cti.gr/ytheod/research/datasets/spatial.html>.
8. K. Park, M. Song, and C. Hwang, "Broadcasting and Prefetching Schemes for Location Dependent Information Services," In *Proc. of W2GIS*, 2004. Lecture Notes Computer Science.
9. O. Kasten, Energy consumption. ETH-Zurich, Swiss Federal Institute of Technology. Available at http://www.inf.ethz.ch/~kasten/research/bathtub/energy_consumption.html.
10. J. Xu, X. Tang, D. L. Lee, and Q. Hu. "Cache Coherency in Location-Dependent Information Services for Mobile Environment," In *Proc. the 1st Int. Conf. on Mobile Data Access*, 1999. Lecture Notes Computer Science.
11. B. Zheng and D. L. Lee, "Semantic Caching in Location-Dependent Query Processing," In *Proc. the 7th International Symposium on Spatial and Temporal Databases*, 2001.
12. T. Camp, J. Boleng and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," In *Wireless Communication and Mobile Computing(WCMC)*, 2002.

Extended Derivation Cube Based View Materialization Selection in Distributed Data Warehouse*

Wei Ye¹, Ning Gu¹, Genxing Yang², and Zhenyu Liu²

¹ Computing and Information Technology Department,
Fudan University, Shanghai, P.R.China
{weiye, ninggu}@fudan.edu.cn

² Shanghai Key Laboratory of Computer Software
Testing and Evaluating, Shanghai, P.R.China
{ygx, lzy}@ssc.stn.sh.cn

Abstract. View materialization is considered to be one of the most efficient ways to speed up decision support process and OLAP queries in data warehouse architecture. There are great varieties of research topics concerning view materialization, such as user query rewrite to transparently direct user query from base table to materialized views, or materialized views update as soon as base table changes, etc. Among most of these topics, a proper selection of views to be materialized is fundamental. While much research work has been done on view materialization selection in the central case, there are still no appropriate solutions to the problem of view selection in distributed data warehouse architecture, which is just the focus of this paper. We model the views in distributed warehouse nodes with derivation cube which is a concept widely used in central data warehouse, and make extensions in order to adapt it to distributed cases. Then, we propose a greedy-based selection algorithm under a storage cost constraint to perform selection process. Finally, a detailed experimental comparison is made to demonstrate the advantage of our solution over simply applying the central methods repeatedly on each warehouse nodes.

1 Introduction and Motivation

A *Data Warehouse* is a repository that integrates information from multiple data sources, which may be heterogeneous, and makes them available for decision support process or OLAP queries [1]. There are several significant benefits of data warehouses: First, they isolate decision support process from the traditional, often on-line database transaction systems, which largely alleviates the burden of database systems, since DSS (Decision Support Systems) does not have so strict requirement on response time as database transaction systems. Second, they extract, integrate and store only “relevant” information from multiple independent information sources. The information is stored in the warehouse ahead of the queries, so user queries are able to

* The work is supported in part by National Natural Science Foundation of China (NSFC) with grant No. 60473124, Shanghai Science and Technology Committee Key Project with Grant No.20051020d1sj05-A and Shanghai Development Foundation of Science and Technology with Grant No.036505001.

be answered just by the data stored in the warehouse and need not be sent to the original sources for execution. Moreover, when the sources are inaccessible due to locks or regular maintenance, data warehouses can still function normally.

The form of data stored in data warehouses is the derived views from information resources, usually database tables (base table), called *materialized views*. Materialized views provide user queries with faster and uniform access to the information integrated from different data sources. Any query that can be rewritten to be answered by materialized views is sped up in such a way. Especially for complicated queries, which often involve large amount of data exchange, the execution time of queries can be dramatically reduced, often several quantitative levels faster. In fact, view materialization is considered to be one of the most efficient ways to speed up decision support process and OLAP queries in data warehouse architecture [2].

There are dozens of research issues concerning view materialization, such as rewriting user query to transparently direct it from base table to materialized views, or materialized views update when modifications of base table occur, etc. Among most of these issues, to select an appropriate set of materialized views is most fundamental. Given a set of queries at the warehouse, we cannot materialize all possible views, as there are constraints on disk space, computation time, or maintenance cost. Therefore, we need to select a proper set of views to materialize in order to minimize the query response time under certain resource constraint.

A number of achievements have been made on the problem of materialized view selection. However, most of them focus on centralized data warehouses. There is still no effective solution for the distributed cases other than repeatedly applying selection algorithms for central cases on each distributed nodes. With the development of business applications, distributed data warehouses are gaining more and more attention. There are various reasons for this: global enterprises often have different local business, which need different local data warehouses; there are too huge data to be stored in one single warehouse; different departments of an enterprise may maintain warehouses of their own... This paper focuses on the problem of materialized view selection in the distributed data warehouse context and provides comprehensive solutions for selection with storage cost constraints.

The rest of the paper is organized as follows: In section2, we introduce distributed view derivation cube to model views and communication network in distributed data warehouse nodes. In section 3, a cost model based on derivation cube is given as quantitative measurement for query response time. Section 4 proposes a greedy-based distributed selection algorithm with a storage cost constraint. A detailed experimental performance analysis is made in section 5 comparing the query response time of our solution with that of simply applying the central methods repeatedly on each warehouse nodes. Finally, we end with related work and some concluding remarks.

2 Distributed Derivation Cube Model

In this section, we introduce distributed derivation cube model as an abstraction of the relation between views and communication network in distributed data warehouse nodes. We start by defining some preliminary concepts.

2.1 Preliminary Definitions

Definition 1: Granularity

The granularity G of a given view v is defined as the orthogonal grouping attributes $\{k_1, \dots, k_n\}$, $\forall i, j, 1 \leq i, j \leq n, i \neq j: k_i \rightarrow k_j$ with \rightarrow being the functional dependence.

Definition 2: View

A view v_N with attributes k_1, \dots, k_n is formalized as $v = (k_{i_1} \dots k_{i_j}, k_{i_{j+1}} \dots k_{i_n})$ with k_{i_1}, \dots, k_{i_j} being orthogonal attribute group, that is $\forall p, q, i_1 \leq p, q \leq i_j, p \neq q: k_p \not\rightarrow k_q$, with N being the warehouse node where v is located.

Obviously, the views in the distributed data warehouse can be essentially represented with their granularities, since those non-orthogonal attributes in the views can be computed from those orthogonal: $v_N = (G)_N$.

Theorem 1: Derivability: $(v_1)_{N1}$ is derivable from $(v_2)_{N2}$ if and only if:

1. Warehouse nodes $N1$ and $N2$ are connected by communication network.
2. The granularity of v_1 is bigger than that of v_2 .

Proof: First, we define a binary relation " \geq ": $ki \geq kj$ means grouping attributes kj can be calculated from ki . This binary relation is apparently a partial order, because: 1) it is antisymmetric, that is, $ki \geq kj$ does not imply $kj \geq ki$; 2) it is transitive, that is, $ki \geq kj$ and $kj \geq km$ imply $ki \geq km$.

It is easy to see that condition 1 is a basic premise for derivation.

Necessity: From condition 2, we know that: for $\forall k_i \in G_1, \exists l_j \in G_2, l_j \geq k_i$, with " \geq " defined above. From definition 2, a view can be characterized by its granularity. So we can see $(v_1)_{N1}$ can be computed from $(v_2)_{N2}$, which proves the proposition.

Adequacy: if $(v_1)_{N1}$ is derivable from $(v_2)_{N2}$, which means $\forall k_i \in v_1, \exists l_j \in v_2: l_j \geq k_i$. Due to the equality between view and its granularity, we have $\forall k_i \in G_1, \exists l_j \in G_2, l_j \geq k_i$. So $G1$ is bigger than $G2$.

By the way, we can see that the derivability relation is also a binary relation satisfying the definition of a partial order. So we use " \Rightarrow " to represent this partial order. $v_1 \Rightarrow v_2$ means the granularity of $v1$ is finer than that of $v2$ and $v2$ can be derived from $v1$.

2.2 Distributed View Derivation Cube

All possible views of a data warehouse, that is, all combinations of grouping attributes[5], forms a directed acyclic graph (DAG) with the edges representing the derivation relationship. Within this DAG, every node stands for one view granularity,

i.e. different views with the same granularity are grouped into one node, which largely reduces the complexity of DAG. A detailed reasoning for derivability is given in [3] [4]. This DAG is the base for distributed view derivation cube.

Although the DAG mentioned above can well capture the view derivation semantics, it is not suitable for distributed cases, since there are extra communication and maintenance costs which are not under consideration in central data warehouse. Therefore, we propose distributed view derivation cube as an extension to the DAG.

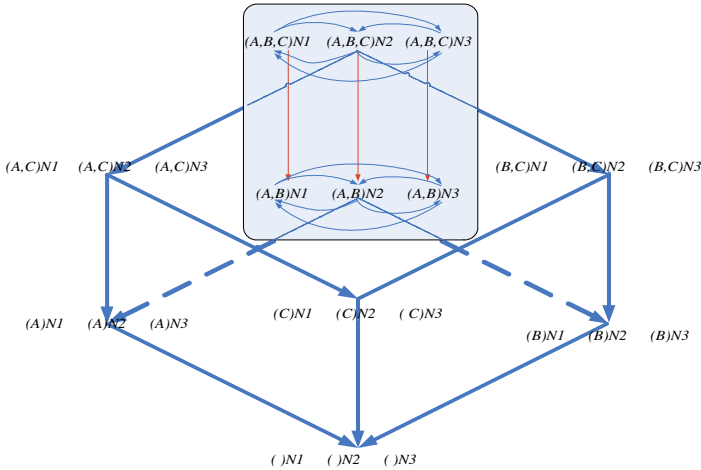


Fig. 1. Distributed View Derivation Cube

As shown in Fig 1. Suppose the base table is (A,B,C) and there are three distributed warehouse nodes $N1, N2, N3$. So there are 8 possible granularities which are shown as 8 nodes of the cube in Fig 1. Every node is split into 3 sub-nodes, each of which denotes the view at one warehouse node. The directed edges between these sub-nodes represent the communication channels. Edges between cube nodes, set red in Fig 1, denote the direct derivation relationship. The derivation relationship between a view v_{Ni} and a dependent view v'_{Nj} on a different warehouse node is not marked out as it can be deduced by the transitiveness of existing relations. For simplicity, only a small part of the cube is presented in detail, as shown in the rectangle in Fig 1.

However, since the communication channels between sub-nodes are bidirectional, cycles appear in the resulting cube, which no longer conforms to the characteristics of DAG. In such cases, selection algorithm can not function properly. To solve this problem, we further modify the cube as illustrated in Fig 2. Each sub-node v_{Ni} of a view v is artificially split into two nodes, an in-node v_{Ni_i} and an out-node v_{Ni_o} respectively. An in-node only have in-edges while an out-node only have out-edges in the sub-graph of a view. It is easy to see that such revision eliminates cycles resulted from communication channels.

Besides conditions in theorem 1, in the context of derivation cube, $v1$ has to be an out-node and $v2$ an in-node if $v_1 \Rightarrow v_2$.

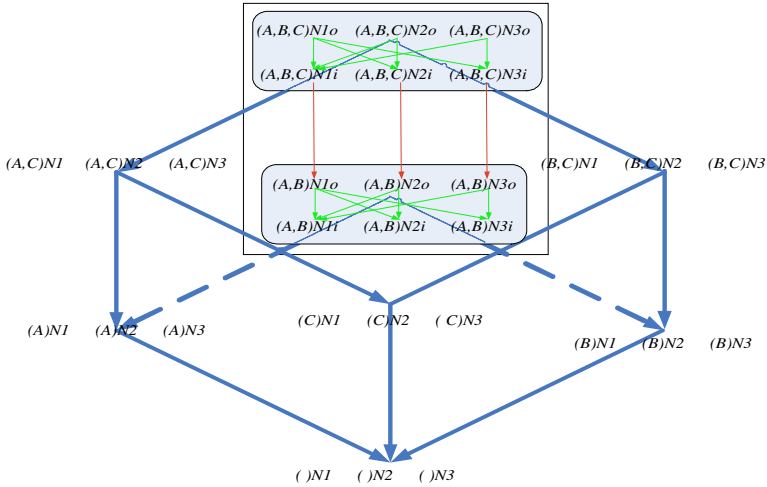


Fig. 2. Modified View Distributed Derivation Cube

Definition 3: Distributed View Derivation Cube

A distributed view derivation cube is defined as a DAG $C(G^n, E)$, G^n is the node set containing all granularity combinations, each of which is divided into n sub-node pairs, an in-node and an out-node, with n denoting the number of distributed warehouse nodes. E is the edge set consisting of derivation edges Ed between nodes of different granularities and communication edges Ec between nodes of the same granularity at different warehouse nodes: $E=\{Ed, Ec\}$.

The edges of the cube are marked with costs: edges in Ed are marked with computation costs for gathering necessary data while those in Ec are tagged with communication cost for data transfer. The cube in Fig 1 assumes a fully connected communication network while in Fig 2, site 2 and 3 are disconnected.

3 The Cost Model

The cost model is fundamental for selection algorithm in section 4. In our approach, we focus on the benefit from the materialization of views. Given a set of queries representing the typical user query behavior, we try to select a set of materialized views with maximized benefit under a storage limitation on each warehouse node.

3.1 The Distributed View Selection Problem

Given a distributed view derivation cube C and a quantity $\{S_1, \dots, S_n\}$ denoting the storage limitations per warehouse node. The *distributed view selection problem* is to select a set of views M , a subset of nodes in C , which minimized the total query response time, under the constraint that the total space occupied by M at i^{th} warehouse node is less than S_i .

More formally, given a distributed view derivation cube C , a set of queries $Q=\{q_1, q_2, \dots, q_l\}$ with querying frequencies $\{f_1, f_2, \dots, f_l\}$ and the storage limitation per

warehouse node $\{S1, \dots, Sn\}$, let $H(q, M)$ denote the cost of answering a query q using the set M of materialized views in the given cube C . The cost $H(q, M)$ consist of two parts: the minimum evaluation costs H_{eval} to answer query q from view $vi \in M$ and the communication costs H_{comm} if the query q is issued at a different warehouse node other than vi is located. The distributed view selection problem is to select a set of views/nodes $M = \{v1, v2, \dots, Vm\}$, that minimized $H(Q, M)$, where

$$H(Q, M) = \sum_{i=1}^l f_i H(q_i, M)$$

3.2 Our Cost Model

3.2.1 Cost Function

In our approach, we consider a linear cost model and assume that evaluation and communication cost functions are dependent respectively on the cardinality of the view $|v|$ and the size of the query result $|q|$. The communication cost functions are weighed by an additional adjusting factor w to tune the ratio of computation to communication costs. So, the overall cost function is:

$$H(q, M) = H_{eval}(q, M) + w * H_{comm}(q, M) = |v_i| + w * |q|$$

The above function is logical because all tuples in v_i have to be read and processed to compute q and the query results have to be transferred from the computing node where v_i is located to the one where the query is initiated.

The complexity of the view selection problem to compute the optimal solution is proved to be NP-hard [3], since there is a straight forward reduction from knapsack problem. So approximations have to be made. In central cases, approximations are mainly based on a greedy approach, which uses a benefit function in order to decide which view to select in each step.

3.2.2 Benefit of a Set of Selected Views

Here, we define the notion of a “benefit” function, which is essential to the development of algorithms presented later. Based on the cost function above, the benefit of Vi with respect to M , an already selected set of materialized views, is denoted by $B(Vi, M)$, and is defined as:

$$B(Vi, M) = H(Q, M) - H(Q, M \cup Vi)$$

The benefit of Vi per unit space with respect to M is $B(Vi, M) / size(Vi)$. If there is no space constraint, all possible views will be materialized to minimize query costs, which is not applicable.

3.2.3 Our Monotonic Benefit Model

Definition 4: Monotonic Property of $B(V, M)$

A benefit function B , which is used to prioritize views for selection, is said to satisfy the monotonicity property for a set of views M with respect to distinct views $V1$ and $V2$ if $B(\{V1, V2\}, M)$ is less than (or equal to) either $B(V1, M)$ or $B(V2, M)$.

In [6], it is shown that the property of monotonicity with regards to the benefit function is a necessary prerequisite to get a reasonably good result using a greedy based approximation. The greedy approach just selecting one single view per iteration may result in an arbitrary bad solution if the monotonicity is not satisfied.

Theorem 2: A benefit function B is monotonic if the following holds:

$$B(V_1 \cup V_2 \dots \cup V_m, M) \leq \sum_{i=1}^m B(V_i, M)$$

The proof is provided in [7]. In the case of view selection problem under space constraint, the query benefit per unit space function $B(V_i, M)/size(V_i)$ satisfies the condition in theorem 2 as it relates just to the size of evaluated views $|V_i|$.

4 Greedy Based Distributed View Selection Algorithm

In this section, we present a greedy based algorithm for solving the view selection problem in distributed data warehouse.

Algorithm 1 : Greedy Algorithm

Input: C , the distributed view derivation cube
 $S=(S_1, \dots, S_n)$, storage limitations for the n warehouse nodes.
Output: M , the set of selected views to be materialized

BEGIN

$M=\{V_0\}$; /* V_0 is the view of base table, with smallest granularity*/

TargetNode= \emptyset ; MaxSpace=0;

While (there is any $S_i > 0$)

$V_t = \emptyset$;

For each in-node V of cube such that $V \in \{C - M\}$

If $B(V, M) > B(V_t, M)$

$V_t = V$;

Endif

Endfor

Let N_i be the warehouse node having the same view granularity as V_t and with largest storage space left

If $(S_i - size(V)) > 0$

$M = M \cup V_{N_i}$;

$S_i = S_i - size(V_t)$;

Else

For $1 \leq i \leq n$ $S_i = 0$;

Endif

Endwhile

Return(M);

END

The main idea of algorithm 1 is as follows: the set of selected views to be materialized M is initialized with base table V_0 . While there is still storage space left at any warehouse node, considering all cube in-nodes not selected to M , the node having the maximum benefit is added to M . Since there may be different warehouse nodes having the same view granularity, we select the view located at the warehouse node with largest storage capacity to add to M . Such iteration continues until no $S_i > 0$.

The running time of this greedy algorithm is $O(mn^2)$, n being the number of nodes in the derivation cube C and m being the number of views chosen for materialization.

5 Compared Analysis of Distributed vs. Local View Selection

In this section, we compare greedy based view materialization selection method for central and distributed data warehouse with an experimental analysis. As illustrated in Fig 2, suppose there are three nodes $N1, N2$ and $N3$ in distributed data warehouse. The ratio of computation cost to communication cost is set to be 1 ($w=1$). The results of queries from query set Q are assumed to cover every view nodes in the cube evenly. The query frequencies f_i are set to be 1. Each node has a storage space of 150.

We first consider the central cases. In Fig 3 is the view DAG for central data warehouse. The first iteration result of single node greedy algorithm is shown in table 1. The node $()$ is selected because of its highest BPUS 7999. Similarly, in the next two round of iteration, (C) and (B) are selected. The total storage space of these three views equals 66 ($1+15+50$). As the storage limitation for one warehouse node is 150, no more views can be selected to materialize.

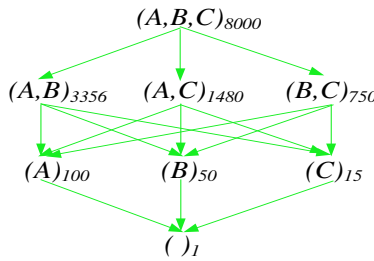


Fig. 3. Single Node View DAG

Table 1. First Iteration Results of Single Node Greedy

	benefit	BPUS
(A,B)	18,576	5.54
(A,C)	26,080	17.62
(B,C)	29,000	38.67
(A)	15,800	158
(B)	15,900	318
(C)	15,970	1064.67
()	7,999	7999

We repeatedly apply the single node algorithm to each node in distributed data warehouse and we get our $M = \{()N1, ()N2, ()N3, (C)N1, (C)N2, (C)N3, (B)N1, (B)N2, (B)N3\}$. The total query costs $H(Q,M)$ amount to 147,570 which consists of $5 * 8,000 + 50 + 15 + 1 = 40,066$ for node N1 and $(8,000 + 3,356 + 1,480 + 750 + 100) * w + 40,066 = 53,752$ for node N2 and N3 with additional communication costs.

Now let's look at the distributed algorithm. Suppose the communication channels between the three warehouse nodes are fully connected, derivation cube is as shown in Fig 2. Selection process is shown in table 2.

The algorithm takes four iterations until the space limitation 150 per node is reached. The M returned by the algorithm is $M = \{()N1, (C)N2, (B)N3, (A)N1\}$. The total query costs $H(Q,M)$ amount to 120,358, which means a cost reduction of $14,7570 - 120,358 = 27212$ (18.44%) is realized. The total storage space used is $1 + 15 + 50 + 100 = 166$ which is much smaller than $(1 + 15 + 50) * 3 = 198$ in the first case.

Based on the above comparison, it is easy to see that our extended derivation cube based approach is superior to original single node greedy approach both in response time and storage cost. The larger the query set Q is, the higher the savings of query costs will be.

Table 2. Selection process for distributed data warehouse nodes

1 st Iteration			
View	Location	benefit	BPUS
(A,B,C)	N2/N3	13,752	1.72
(A,B)	N1/N2/N3	55,728	16.61
(A,C)	N1/N2/N3	78,240	52.86
(B,C)	N1/N2/N3	87,000	116
(A)	N1/N2/N3	47,400	474
(B)	N1/N2/N3	47,700	954
(C)	N1/N2/N3	47,910	3,194
()	N1/N2/N3	23,997	23,997

2 nd Iteration			
View	Location	benefit	BPUS
(A,B,C)	N2/N3	13,751	1.72
(A,B)	N1/N2/N3	36,467	10.87
(A,C)	N1/N2/N3	49,362	33.35
(B,C)	N1/N2/N3	68,250	91
(A)	N1/N2/N3	23,700	237
(B)	N1/N2/N3	23,850	477
(C)	N1/N2/N3	23,955	1,597
()	N2/N3	1	1

Table 2. *Continued.*

3 rd Iteration			
View	Location	benefit	BPUS
(A,B,C)	N2/N3	13,736	1.72
(A,B)	N1/N2/N3	36,467	10.87
(A,C)	N1/N2/N3	30,218	20.41
(B,C)	N1/N2/N3	46,500	62
(A)	N1/N2/N3	23,700	237
(B)	N1/N2/N3	23,850	477
(C)	N1/N2/N3	15	1
()	N2/N3	1	1
4 th Iteration			
View	Location	benefit	BPUS
(A,B,C)	N2/N3	13,686	1.71
(A,B)	N1/N2/N3	22,378	6.67
(A,C)	N1/N2/N3	30,218	20.41
(B,C)	N1/N2/N3	23,250	31
(A)	N1/N2/N3	23,700	237
(B)	N1/N2/N3	50	1
(C)	N1/N2/N3	15	1
()	N2/N3	1	1

6 Related Work

In the initial research done on the view selection problem in central data warehouses, Harinarayan et al. [8] presented greedy based algorithms for solving view selection problem in data cubes under a storage constraint. In their work, a data cube is a special purpose warehouse where there are only queries with aggregations over the base table. Gupta et al. extended their work to include indexes in [9]. The approach presented in this paper is the generalization and extension of the work in [8] to distributed cases.

Negative theoretical results on the view selection problem are also obtained. In the work of Chirkova et al. [10], the complexity of the view selection problem to compute the optimal solution is proved to be NP-hard. Karloff and Mihail [11] show that the view selection problem trying to get the optimized query cost is inapproximable for general partial orders. The result was later extended in [10][12]. In both papers, the authors assume that the input for algorithm only consist of the database schema and workload queries, while in our approach the algorithm starts with the derivation cube.

There are also some jobs done on improving actual selection performance by developing heuristics [13][14][15][16], Most of the work has developed different infrastructure and heuristics algorithms for view selection. However, these approaches

are either exhaustive searches or without any performance guarantees on the quality of the solution delivered.

Another group of work focuses on the maintenance of the materialized views, hoping to minimize view synchronization costs. This problem is more complicated since the benefit function with respect to the maintenance costs exhibits non-monotonic properties. [6] did some sensible work in this context.

[17][18][19] apply the idea of caching the result of user queries to speed up similar queries of other users. Determinative algorithms are developed to decide whether a query result should be put in cache or not. Meanwhile, replacement strategies have to choose those temporarily unused results out of cache in favor of the new result set.

No proper solutions specially developed for view selection in distributed data warehouse are found in related work.

7 Concluding Remarks

A data warehouse is built for better information integration and decision support process. The selection of views to be materialized is one of the most fundamental issues in designing a data warehouse. While lots of work has been done on the view selection in central data warehouses, no proper solution is given in distributed cases, which is just the focus of this paper. The techniques developed in this article offer significant insights into the nature of the view selection problem in distributed data warehouses.

In particular, we model the views in distributed warehouse nodes with derivation cube which is a concept extended from the data cube in central data warehouse, and make extensions in order to adapt it to distributed cases. Then, a greedy-based selection algorithm to perform selection process under a storage cost constraint is presented. A detailed comparison analysis is made to demonstrate the advantage of our solution over simply applying the central methods repeatedly on each warehouse nodes. Statistical data shows that our approach is far better both in response time and storage space costs.

There are still a lot of important issues in the context of view selection in distributed data warehouses. Future work will focus on improving our algorithm for further approximation to the optimal set to be materialized and decrease its running time. A more typical and complex performance analysis will be provided to make it more persuasive. Moreover, aggregations queries over more than one base table and queries other than aggregations will be taken into account to make our solution applicable for more common occasions.

References

1. R.Alhajj and A.Elnagar: Incremental Materialization of Object-Oriented Views, *Data and Knowledge Eng.*, vol.29, no.2, pp.121-145, 1999.
2. R. Kimball: *The Data Warehouse Toolkit*. John Wiley & Sons, Inc., 1996.
3. Gupta, H.: Selection of Views to Materialize in a Data Warehouse. In: *Proceedings of the 6th International Conference on Database Theory , ICDT'99, Delhi, January 8-10, 1999*

4. Albrecht, J.; Guenzel, H.; Lehner, L.: Set-Derivability of Multidimensional Aggregates. In: Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery, DAWAK'01, Florence, Italy, August 30-September 1, 2001
5. Gray, J.; Bosworth, A.; Layman, A.; Pirahesh, H.: Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total. In: Proceedings of the 12th International Conference on Data Engineering, ICDE'98, New Orleans (LA), U.S.A., February 26-March 1, 1998
6. Gupta, H.; Mumick, I.: Selection of Views to Materialize Under a Maintenance Cost Constraint. In: Proceedings of the 7th International Conference on Database Theory, ICDT'01, Jerusalem, Israel, January 10-12, 2001
7. Gupta, H.: Selection of Views to Materialize in a Data Warehouse. In: IEEE Trans. on Knowledge and Data Engineering, Vol. 17, No. 1, Jan. 2005 24-43.
8. V. Harinarayan, A. Rajaraman, and J. Ullman: Implementing Data Cubes Efficiently, Proc. ACM SIGMOD Int'l Conf. Management of Data, 1998.
9. H. Gupta, V. Harinarayan, A. Rajaraman, and J. Ullman: Index Selection in OLAP, Proc. Int'l Conf. Data Engineering, 1998.
10. R. Chirkova, A. Halevy, and D. Suciu: A Formal Perspective on the View Selection Problem, Proc. Int'l Conf. on Very Large Database Systems, 2001.
11. H. Karloff and M. Mihail: On the Complexity of the View-Selection Problem, Proc. Symp. on Principles of Database Systems (PODS), 1999.
12. R. Chirkova: The View Selection Problem Has an Exponential Bound for Conjunctive Queries and Views, Proc. ACM Symp. on Principles of Database Systems, 2004.
13. J. Yang, K. Karlapalem, and Q. Li: Algorithms for Materialized View Design in Data Warehousing Environment, Proc. Int'l Conf. on Very Large Database Systems, 1999.
14. E. Baralis, S. Paraboschi, and E. Teniente: Materialized View Selection in a Multidimensional Database, Proc. Int'l Conf. on Very Large Database Systems, 1997.
15. D. Theodoratos and T. Sellis: Data Warehouse Configuration, Proc. Int'l Conf. on Very Large Database Systems, 1999.
16. A. Shukla, P. Deshpande, and J. Naughton: Materialized View Selection for Multidimensional Datasets, Proc. Int'l Conf. on Very Large Database Systems, 2001.
17. Deshpande, P.; Ramasamy, K.; Shukla, A.; Naughton, J.: Caching Multidimensional Queries Using Chunks, in: 27th International Conference on the Management of Data, SIGMOD'98, Seattle, USA, June 2-4, 1998
18. Scheuermann, P.; Shim, J.; Vingralek, R.: WATCHMAN: A Data Warehouse Intelligent Cache Manager, in: 22nd International Conference on Very Large Data Bases, VLDB'99, Bombay, India, September 3-6, 1999
19. Kotidis, Y.; Roussopoulos, N.: DynaMat: A Dynamic View Management System for Data Warehouses. In: Proceedings ACM International Conference on Management of Data, SIGMOD'02, Philadelphia (PA), U.S.A., June 1-3, 2002

A Framework of HTML Content Selection and Adaptive Delivery*

Chen Ding¹, Shutao Zhang², and Chi-Hung Chi³

¹ School of Computer Science, Ryerson University, Canada

² School of Computing, National University of Singapore, Singapore

³ School of Software, Tsinghua University, Beijing, China 100084
chichihung@mail.tsinghua.edu.cn

Abstract. In this paper, we propose a content selection and adaptive delivery framework to adapt content delivery to different client capabilities and preferences. The framework provides language support to content publishers with XML-based markup language to describe the content and control the transformation and delivery process. Automatic content classification and selective delivery based on match between content class and user capability or preference are also implemented. Thus, when content descriptions from server are not available, adaptive content delivery is still possible. The framework is generic and scalable. Performance studies on content classification and selection model show that it is effective in saving network resources and improving clients' access experiences.

1 Introduction

Today's Internet computing environment has changed a lot since its first appearance. It is pervasive and ubiquitous, which means more and more devices such as personal digital assistants (PDAs), Pocket PCs, set-top boxes and cellular phones can access the Internet, not just personal PCs as it was initially intended for. As pointed out in [7], the number of wireless Internet clients is predicted to reach 48% of all Internet clients by the end of 2005. All the devices, both wired and wireless, are diverse in hardware (processing power, memory size, I/O capability, screen size, etc.), software (operating system, browser, audio-visual applications, etc.) and network access methods (communication media and speed). It is inappropriate if content publishers deliver the same content to every client. It will be better if content delivery could be adapted to different client capabilities.

Since the Internet has become a venue for companies, organizations, governments, or personal users to publish information, to conduct business, or to provide service, user-orientation or personalization has been more and more emphasized. Users are diverse, and thus they have different preferences and requirements on the content they want to receive from the Internet, such as language preference, interest on advertisement, privacy preservation, etc. It will be better if content delivery could be adapted to different client preferences.

* This research is supported by the funding 2004CB719400 of China.

On the Internet, World Wide Web is the most accessed application, while on the web, HTML is the primary language used for building the web site and represent 20% of total web traffic [3]. A high percentage of web traffic is coming from multimedia content, which is usually embedded in or linked from a HTML page. Currently there are many adaptive content delivery solutions of dissolving mismatch between content and capability by transcoding multimedia content to a more appropriate representation form. For example, images can be reduced in size to fit in a smaller screen. There are several limitations in these solutions.

- Many of them just focused on one type of multimedia content, e.g. image, video or audio, and how to transform them into a more appropriate form. They didn't look at HTML document as a whole, which includes both HTML page and all embedded multimedia objects. Therefore, they couldn't make use of the relationship between the container page and component objects to remove the content not needed by users.
- Some solutions made transcoding decisions in web intermediaries (e.g. proxies) based on implicit information such as HTTP Content-Type header. Proxy-based solution is good because it can improve the performance and scaling, offload the original web server, and save the work of web publishers of providing transcoding guidance. However, ambiguous information and simple one-for-all mechanism might result in some bad transcoding decisions.
- On the contrary, some solutions either did transcoding on server side or relied on instructions provided by servers. The accuracy could be improved. But there are some other problems, e.g. computation load on server is increased, more storage spaces are required for different versions of content, web publishers should put extra effort on annotate the content or write control applets. And another problem for this approach is that it is not feasible for existing web pages.

In this paper, we focus on HTML content, which includes both HTML page and all the embedded objects. We work on the container first, which is HTML page itself, and then on all the embedded multimedia objects. We are trying to put up with a solution which is generic and overcomes above-mentioned problems.

Most of HTML tags are designed to help web publishers present their information in a certain format, while some of them are not. For example, values of "keywords" and "description" attribute in "meta" tag are used by search engines to identify keywords in the page, while users may not care about them. Edge Side Includes (ESI) [6] language markups may be inserted into HTML pages to facilitate content caching and selection, and they are more for web intermediaries instead of end users. Usually, web clients are only interested in content which could be presented through browsers. So, if HTML content is not designed for presentation purpose, it doesn't have to be delivered to clients.

Based on this observation, we propose a content selection framework to deliver only the content which users want and which devices can handle. The content classification is based on heuristics instead of human-edited descriptions. It is autonomous, and there is no overhead on web publishers, both in server processing power and in manpower. It is especially suitable for those existing HTML documents.

No matter how complicated of those heuristics, there is always a risk of losing important information or losing opportunities for aggressive transcoding. In order to have a more complete solution, we propose a generic framework for content selection and delivery. First, our framework supports our previously proposed Content Markup Language (CML) [5] with which server can control the transformation and delivery process. Second, we implement automatic content classification and selectively deliver content based on its class and user capability or preference. Thus, when server instructions or annotations are not available, adaptive content delivery is still possible. Third, content selection and transformation are performed on proxy server, which has many benefits as claimed in [13]. To enable cooperation with other proxies and address data integrity concerns, our proposed language model also provides a way for proxies to leave their traces. Currently, in our framework, we assume clients express their preferences or device capabilities explicitly using some existing technologies [4] [19].

Key component in this framework is automatic content classification and selection model. The reason for such a focus is that, from experiences we learned from today's Internet, solution which is simple and automatic, usually can be deployed more quickly and by larger user bases. The proposed framework also incorporates the strengths of current solutions.

The rest of the paper is organized as follows. In section 2, we review related works on adaptive content delivery. In section 3, we describe the overall architecture of our content selection and delivery framework. Section 4 details how the framework can be deployed and some implementation concerns. Finally in section 5, we conclude our work and list several future works.

There are many different definitions for HTML document and HTML content. Throughout this paper, we use HTML document to refer to a HTML page and its embedded objects, HTML content to refer to any content included in a HTML document, and HTML object to refer to individual object in a HTML document.

2 Related Works

With diversified and ever changing client capabilities and preferences, it is a big challenge for web content publishers with one source to offer the most cost-effective, best-fit content to its global users. To enable web content publishers to provide customized content to different clients, researchers have proposed general frameworks in which web content can be modified on the way from servers to clients, including Internet Content Adaptation Protocol (ICAP) [11] and Open Pluggable Edge Services (OPES) [15]. ICAP provides a framework in which almost any services for adaptive web content delivery can be implemented. ICAP clients can pass HTTP messages to ICAP servers for content modification.

Besides the framework, there are extensive researches on content transformation or adaptation. Web content transformation can be performed on web servers, web intermediaries (i.e. proxies), or web clients. Many of the solutions are proxy based [1] [8] [17] and proxy-based content transformation is usually called transcoding. Fox et al. [8] developed a system to distill or transcode images when they pass through a proxy, in order to reduce the bandwidth requirements. Chandra and Ellis [1] presented

techniques of quantifying the quality-versus-size tradeoff characteristics for transcoding JPEG images. Spyglass Prism [17] is a commercial transcoding proxy. It compresses JPEG images and changes the size and color depth of images to improve the download time. There are also some approaches trying to address the issue of content transformation on the server side. They either provide external content annotations or give guidance on how to process the web content. Hori et al. [10] proposed to annotate HTML pages to guide the transcoding procedure. Mogul et al. [12] [13] proposed a server directed transcoding scheme.

Most of content transformation solutions focus on one or two types of multimedia objects without looking into the whole web document. InfoPyramid [14] [16] actually considered the web document as a whole, which is similar to our approach. It is a representation scheme for handling the Internet content (text, image, audio and video) hierarchically along the dimension of fidelity (in different qualities but in the same media type) and modality (in different media types). It also provided a model to generate and organize different versions of Internet content and then deliver suitable content based on client capabilities. However, it relied on content descriptions (in XML) to determine their resource requirements, and thus it may not work for those existing HTML documents. The need to generate and store different versions of content could also be a scalability problem.

There are some commercial software tools customizing HTML content according to different clients' needs. Examples include WebOverDrive [20], HTML Optimizer X [9], etc. They can "compress" HTML content by filtering out some parts of content that clients are not interested in. Usually this kind of tools only removes blank spaces, new lines and comments. There is no sophisticated classification method on HTML content, and also they only focus on the HTML page itself.

To deliver the best-fit content to clients, we need descriptions about the client's preference and capabilities in the first place. W3C has proposed the Composite Capability and Preference Profile (CC/PP) [4] to achieve this goal. Wireless Application Protocol (WAP) Forum has proposed a similar approach called User Agent Profile (UAProf) [19] to handle clients' descriptions. Both CC/PP and UAProf are based on Resource Description Framework (RDF) and their goals are describing and managing software and hardware profiles. In this study, we will use CC/PP or UAProf for client side descriptions.

3 Content Selection and Delivery Framework

The general process of content selection and adaptive delivery is illustrated in Figure 1. When a client wants to access certain HTML content from a server, it sends a request to the server together with client descriptions on its preferences and hardware capability constraints. For example, the client may not be interested in, or not be able to handle images and video, but may be particularly interested in audio. Upon receiving the request, the web server retrieves the content and passes it to a content selector, together with client descriptions, and content descriptions if they are provided by content publishers. The content selector then selects HTML content based on both client and content descriptions, transforms content when necessary, and passes the selected and transformed content to the client.

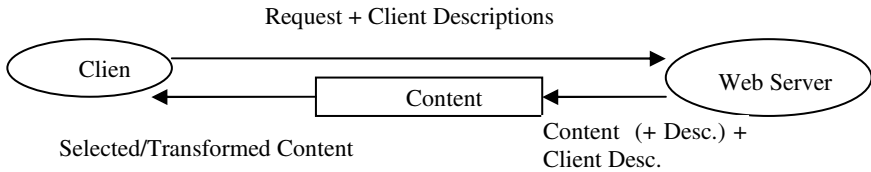


Fig. 1. General Settings of the Framework

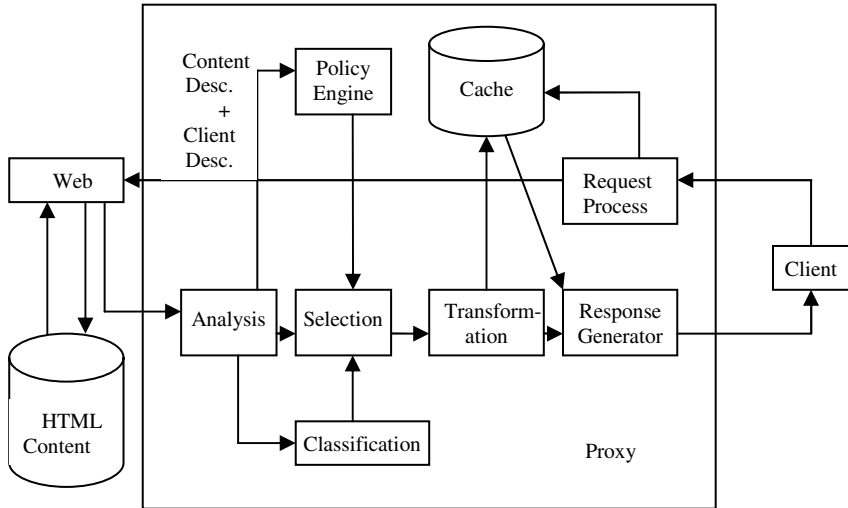


Fig. 2. Content Selection and Delivery Framework

Note that the notion of a content selector is very general. It could be implemented within a web server, as a service provided by a third-party, as an add-on function of a proxy server, or even in the form of a personal firewall. Figure 2 illustrates one possibility when it is implemented in a proxy server. In the remaining of the paper, we assume that content selection is implemented in proxy server. We will explain the reason for such a choice in section 6.

In our framework, content publishers are given choices of whether to control the content selection and delivery process or not. If they want to control the process, they can provide content descriptions of importance level, different attributes and transcoding instructions on each fragment of HTML content. Sometimes, they can also provide partial descriptions or descriptions on part of the web site. Given this kind of flexibilities, content publishers can find a tradeoff between more workload of adding extra descriptions and more control on content selection and transformation.

As we mentioned before, we assume that clients can use existing technologies such as CC/PP or UAProf to describe their preferences or device capabilities. When clients send the request, they also attach client descriptions. The proxy-based implementation of our framework as the one shown in Figure 2 can search the cache first trying to find a copy of content with matched client descriptions. If search fails,

proxy sends request to web server. When receiving the content from server, it analyzes the content. If content descriptions are attached, content can be passed to selection module directly. Otherwise, content fragments should be automatically classified into different classes and then be passed to selection module. Depending on client descriptions and content descriptions (or content classes), policy engine directs the process of content selection and transformation. The selected and transformed content can be sent back to client, and in the mean time, also be saved in local caches.

In the following sections, we give more detailed description on content markup language model, and several concerns in the actual deployment of the framework. More detailed description about the content selection model and its markup language can be found in [5].

4 Deployment of Framework

4.1 General Considerations

There are several concerns in the implementation of our framework. First is the scalability. It should be able to handle the increase of client numbers. Second is to save network resources. It is one of the important goals of designing this framework. Third is transparent content selection and transformation. Clients only need to express their preferences and device capabilities. No change on the client side software is required. And they don't need to send special request or use special protocols. In our current system, only content selection is implemented. But with the proposed framework, it is easy to incorporate transformation modules. The rest of the discussion will focus on our current implementation and its underlying reasons.

We have a few options on who makes content selection and transformation decisions, including web servers, proxies and clients. There are different benefits and constraints for different options.

For web servers, they host HTML documents with content descriptions. If they are doing content selection, the benefit is that they don't have to transfer HTML documents with all content descriptions, but only contents after selection need to be delivered to clients. However, they have to get client descriptions (preferences, capabilities) to make a decision on content selection. Clients can be from anywhere in the world and using any kind of devices to access the web site, and their preferences and device capabilities vary a lot. When the number of clients increases, the server has to increase its capability (processing power, network bandwidth) to handle more requests. So this solution has a problem of scalability.

If proxies are doing content selection, they also have advantages and disadvantages. The disadvantage is that additional bandwidth is required to transfer content descriptions from servers. The advantage is that they are distributed among the whole network; each of them only handles a small portion of clients accessing the web site. So scalability is not an issue for proxies. Since there are much more clients than proxies, so the saving in network bandwidth from proxies to clients is much bigger than bandwidth required from servers to proxies. Additionally, proxies can cache original HTML documents (with content descriptions) as well as client descriptions to reuse them later, thus further save the time and network resources to transfer all the information to clients. We do not consider the possibility of performing content selection on the client side as it is directly against our design objectives.

There is another alternative solution, i.e. web servers instruct content selection process and proxies actually make decisions based on server instructions, much like server directed transcoding [13]. This way, server's workload is not that severe, and it still can control what content be selected. But this alternative has its own problems. First, we need additional support from the web architecture (protocols, bandwidth, etc.) to pass server's instructions to proxies. This will increase the complexity of the deployment. Due to the distributed nature of network, it will make the deployment even harder. Another constraint is that servers cannot ensure that proxies will follow their instructions. As each proxy has its own configuration and policies, it may or may not follow instructions from servers.

From above analyses, we believe that proxies are most suitable for doing content selections. Therefore, we use proxy-based approach in our implementation. Before we look into detailed operations, we will make the following assumptions:

Data Integrity. Since proxies can modify a message on the way from a server to a client, we assume the client can always trust the received message. In other words, the data is preserved for their intended use, which includes content transformation by the authorized, delegated web proxies.

Correct Proxy Delegation. We assume only delegated proxies will perform authorized operations on content and traces of their operations are marked correctly. And no malicious proxies can modify the content by performing unauthorized operations.

4.2 Server Operations

We server is only an information provider but not a decision maker. It needs to store the content as well as its descriptions if they are provided by content authors. Content descriptions are generated offline by content authors. They can use CML language tool to divide HTML pages into fragments and provide descriptions for these fragments.

Since content descriptions are embedded into HTML documents, when a request for a HTML document arrives, there are two possibilities – request for a pure HTML document, or request for a HTML document with content descriptions. The former could be from a proxy or client which is not aware of content selection or does not understand it. The latter could be from a proxy which has implemented our proposed selection and delivery functionalities. Server has to differentiate between these two possibilities and respond accordingly. We make use of HTTP/1.1 “Accept” header. We invent a new field value “content-selection” for the header. When clients/proxies need content descriptions from the server, add “Accept: content-selection” in the HTTP header field in request. When the server sends response to those who need content descriptions, it includes “207 content-selection” if it is successful.

4.3 Proxy Operations

When proxies act as a content selector, they should implement two types of operations: content selection and reassembly, as well as operation trace marking. The latter is to promote cooperation among proxies on content selection, and also for the possible data integrity checking.

4.3.1 Content Selection and Reassembly

In order to perform content selection for clients, proxies have to gather information from both web servers and clients. When sending requests to web servers, proxies need to include “Accept: content-selection” in the HTTP header field to indicate it needs content descriptions. For clients’ descriptions (preferences and capabilities), there are number of ways, such as HTTP headers or other markup languages including CC/PP and UAProf. After getting HTML content with descriptions and client descriptions, proxies can start to perform content selections accordingly. All heuristics and rules are implemented in a policy engine. As we discussed before, it only includes rules for selection currently. It will be easy if later we include transformation rules. If client descriptions are missing, all content will be selected. If content descriptions are missing, automatic classification can help make selection decision.

After content selection is finished, proxies can reassemble the content to form a complete HTML document and respond to clients with all the traces of operations to notify clients on what operations has been done.

4.3.2 Operation Trace Marking

When web content is delivered from server to client, there can be many proxies on the way performing content selections. It is important to know what others have done on the content. This requires proxies to leave traces of their operation in HTML content. Leaving traces of operations has two requirements. First, the proxy doing the operation should declare itself in the trace. Second, it should describe what operations are performed and what property of the (modified) content is.

In general, traces of an operation are marked in between *<property>* and *</property>* tag pairs. If there are multiple operations performed on the content, the sequence to mark up these operations is the same as the sequence to perform these operations. So when an operation is finished, its traces will always be appended to the end. It can preserve the sequence of operations performed. The operation traces are illustrated in Figure 3.

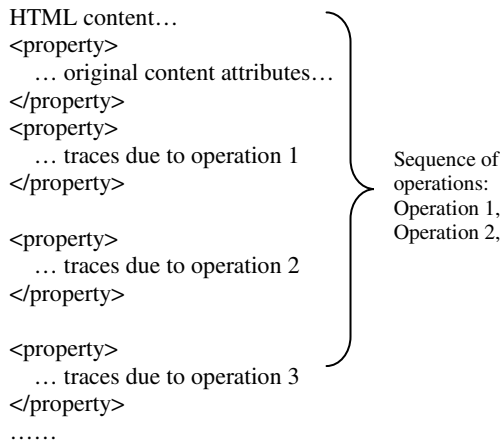


Fig. 3. Operation Traces General Format

4.4 Client Operations

Clients need to send request for HTML documents and express their preferences and device capabilities with the request. If a client is only interested in HTML content, then a normal HTTP request is enough. If a client is both interested in HTML content and content descriptions (possibly with operation traces), it can add the “Accept: content-selection” header to the HTTP request.

To send out preferences and capabilities, there are a number of ways to do it. Clients can utilize HTTP header fields like “Accept-language”, “Accept-Encoding” to express some of their preferences. Or they can rely on existing standards such as CC/PP or UAProf to express their preferences and device capabilities in greater detail. They can also use existing profile repository [18], like some of WAP phones.

5 Conclusion

In this paper, we propose a content selection and adaptive delivery framework for today’s pervasive and personalized web environment. The framework incorporates strengths from previous researches (e.g. providing server control on delivery process), and also has its unique contributions. First, it has an automatic content classification and selection model, in which content can be classified based on its functionality, matched with user preferences, and then selected, transformed and delivered to users. It is more accurate than one-for-all mechanism and has less reliance on server instructions. Second, the language model not only supports servers to provide content descriptions or guidance on transformation, but also supports proxies to leave their traces, which improves data integrity and cooperation among multiple proxies. Third, the framework is generic and flexible. It works in many different situations, for existing HTML documents or for new web site, with or without server instructions, with a complicated transformation rule engine or simply making a decision between discarding and retaining. A HTTP header extension is also introduced to keep content selection transparent for parties which do not understand the selection model.

Since the proposed framework is generic, implementations on each component could vary from system to system. Currently we haven’t implemented the whole framework yet. Automatic content classification and selection has been tested and performance improvement has been verified [5], while content transcoding hasn’t been integrated to test the overall performance. That will be one of our future works.

There are several other works we want to continue in the future. First, currently we only implement two transformation actions. We will try to build a rule engine with more complicated rules to trigger more transformation actions such as image transcoding. Second, we would like to have a user interface for updating the rule engine. It would be easier to add in new rules with the interface when new client descriptions or new rules or new transformation algorithms come out. It is also easier for deleting, updating and other maintenance. Finally, although we have some mechanism for proxies to leave traces, it is not enough to address issues of data integrity and proxy delegation. Since there are existing solutions [2], they can be added to the framework to provide a more robust framework.

References

- [1] S. Chandra, C. S. Ellis, JPEG Compression Metric as a Quality Aware Image Transcoding, In *Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems*, 1999.
- [2] Chi-Hung Chi, Xiao-Yan Yu, Wenjie Zhang, Chen Ding, and Weng-Fai Wong, Data Integrity Framework and Language Support for Active Web Intermediaries, In *Proceedings of 9th International Workshop on Web Caching and Content Distribution (WCW)*, 2004.
- [3] ClickZ Internet Statistics and Demographics, <http://www.clickz.com/stats/>.
- [4] Composite Capability and Preference Profile, <http://www.w3.org/Mobile/CCPP/>.
- [5] Chen Ding, Shutao Zhang, Chi-Hung Chi, "Content Selection Model for Adaptive Content Delivery," Internal Report, National University of Singapore, 2005.
- [6] ESI language specification 1.0, <http://www.esi.org>, 2000.
- [7] eTForecasts, <http://www.etforecasts.com/pr/pr0402b.htm>.
- [8] A. Fox, S. D. Gribble, E. A. Brewer, and E. Amir, Adapting to Network and Client Variability via On-Demand Dynamic Distillation, In *Proceedings of 7th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 1996.
- [9] HTML Optimizer X, <http://www.macupdate.com/info.php/id/7935>.
- [10] M. Hori, G. Kondoh, K. Ono, S. Hirose, and S. Singhal, Annotation-Based Web Content Transcoding, In *Proceedings of The 9th WWW Conference*, 2000.
- [11] Internet Content Adaptation Protocol (I-CAP), <http://www.i-cap.org>.
- [12] B. Knutsson, H. H. Lu, and J. C. Mogul, Architecture and Pragmatics of Server directed transcoding, In *Proceedings of the 7th International Workshop on Web Content Caching and Distribution*, 2002.
- [13] J. C. Mogul, Server Directed Transcoding, In *Computer Communications* 24(2):155-162, 2001.
- [14] R. Mohan, J. R. Smith, and C. S. Li, Adapting Multimedia Internet Content for Universal Access, In *IEEE Transactions on Multimedia*, 1(1):104–114, 1999.
- [15] Open Pluggable Edge Service (OPES), <http://www.ietf-opes.org>.
- [16] J. R. Smith, R. Mohan, and C. S. Li, Transcoding Internet Content for Heterogeneous Client Devices, In *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, 1998.
- [17] Spyglass-Prism, <http://www.opentv.com/support/primer/prism.htm>.
- [18] UAProf profile repository, http://w3development.de/rdf/uaprof_repository/.
- [19] User Agent Profile, WAP forum, <http://www.wapforum.org/what/technical/SPEC-UAProf-19991110.pdf>.
- [20] WebOverDrive, <http://weboverdrive.glopro.com/>.

An Improved Multi-stage (t, n) -Threshold Secret Sharing Scheme

Hui-Xian Li^{1,2}, Chun-Tian Cheng², and Liao-Jun Pang³

¹ School of Electronic and Information Engineering, Dalian University of Technology,
11 60 24 Dalian, China
hxli@student.dlut.edu.cn

² Institute of Hydroinformatics, Dalian University of Technology,
11 60 24 Dalian, China
ctcheng@dlut.edu.cn

³ National Key Laboratory of Integrated Service Networks, Xidian University,
71 00 71 Xi'an, China
pangliaojun_x@etang.com

Abstract. Recently, Chang *et al.* proposed a multi-stage (t, n) -threshold secret sharing scheme based on the one-way function. For k secrets shared among n participants, each participant has to keep one secret shadow; but there are a total of kn public values. In this paper, a new multi-stage (t, n) -threshold secret sharing scheme was proposed, which is an alternative implementation of Chang *et al.*'s scheme. Each participant needs to keep only one secret shadow in sharing multiple secrets without updating each participant's secret shadow. For k secrets shared among n participants, there are only a total of $k(n-t+1)$ public values. Analyses show that the proposed scheme is a computationally secure and efficient scheme. The implementation of the proposed scheme becomes very attractive, especially when the threshold value t is very close to the number of participants n .

1 Introduction

A secret sharing scheme is a technique to share a secret among a group of participants. It plays an important role in protecting important information from being lost, destroyed, modified, or into wrong hands. Secret sharing has been an important branch of modern cryptography [1], [2]. A secret sharing scheme mainly includes a pair of algorithms [3]: a secret-distribution algorithm and a secret-reconstruction algorithm. A trusted dealer (secret holder) computes n shadows (also called sub-secrets or shares) of a shared secret via the secret-distribution algorithm and gives them to n participants securely, respectively. The secret-reconstruction algorithm is executed by a qualified subset of participants who, by putting together their own secret shadows, can therefore reconstruct the shared secret. The first secret sharing scheme is the (t, n) -threshold scheme, which was introduced by Shamir [4] and Blakley [5] independently, respectively. In the (t, n) -threshold secret sharing scheme, a trusted dealer divides a secret into n shadows, and then he distributes these shadows to n participants. Any subset of size at least t participants can reconstruct the shared secret, but $(t-1)$ or fewer participants can obtain nothing about the secret.

There are mainly two categories of secret sharing schemes according to the order of secret reconstruction, the general secret sharing scheme and the multi-stage secret sharing scheme. In the former, the secret reconstruction can be executed in any order, and multiple secrets can be reconstructed simultaneously in a secret sharing session, such as schemes [6], [7], [8]. In the latter, the secret reconstruction must be executed in a predefined order, such as [9], [10], [11]. In the real world applications, the multi-stage secret sharing is very practical. For example [11], there may be a security system of bank's confidential database where one must pass through k checkpoints before the database can be accessed. To distribute the power of a single authority and the security policy, the checkpoints should be opened and passed in sequence by at least t participants together. If the number of participants is less than t or the checkpoints (secrets) do not follow the proper order, it will harm the security of the system. So it is very significant to study the multi-stage secret sharing.

In 1994, He and Dawson [9] proposed a multi-stage (t, n) -threshold secret sharing scheme to share multiple secrets based on the one-way function. They used the public shift technique to obtain each participant's true shadow and the successive application of a one-way function to make secrets reconstructed stage-by-stage in a special order. Later, Harn [10] proposed an alternative scheme, which has the same capability as He and Dawson's scheme. Recently, Chang *et al.* [11] point out that both He and Dawson's and Harn's schemes are one-time use schemes [12], that is to say, when some particular secrets have been reconstructed, it is required that the dealer redistributes fresh shadows to every participant. Then they proposed a new multi-stage (t, n) -threshold scheme [11] based on the one-way function. They claimed that their scheme belongs to the multi-use scheme [12], that is to say, when some particular secrets have been reconstructed, it is not required that the dealer redistributes fresh shadows to every participant, and each participant needs to keep only one shadow. However, Chang *et al.*'s scheme used kn public values to share k secrets. The number of public values is one of the important parameters that determine the performance of a scheme, because it affects the storage and communication complexity of the scheme [7], [13]. Therefore, the large number of the public values may affect the performance of Chang *et al.*'s scheme. Motivated by these concerns, we made an improvement on Chang *et al.*'s scheme, and proposed an alternative implementation of Chang *et al.*'s scheme. The proposed scheme has the same merits as Chang *et al.*'s scheme. What's more, for k secrets shared among n participants, only a total of $k(n-t+1)$ public values are needed in our scheme, which are fewer than that in Chang *et al.*'s scheme. Our scheme becomes very attractive, especially when the threshold value t is very close to the number of participants n .

The rest of this paper is organized as follows. In Section 2, we shall briefly review Chang *et al.*'s scheme. In Section 3, we shall present our multi-stage (t, n) -threshold secret sharing scheme. In Section 4, we shall make an analysis and discussion about the proposed scheme. Finally, we shall give conclusions in Section 5.

2 Review of Chang *et al.*'s Scheme

Before presenting Chang *et al.*'s scheme, we introduce the definition of the one-way function [9], which will be also used in the proposed scheme.

Definition 1 (One-way function). Let $GF(p)$ denote a finite field, and let $f: GF(p) \rightarrow GF(p)$ be a one-way function. For any m and any non-negative integer k , $f^k(m)$ denotes k successive applications f to m , i.e., $f^0(m) = m, f^k(m) = f(f^{k-1}(m))$ for $k > 0$.

Now we review Chang *et al.*'s scheme briefly, and for more details, readers may refer to the reference [11]. Suppose there are k secrets $s_i (i=1, 2, \dots, k)$ to be shared among n participants. Let x_1, x_2, \dots, x_n be n distinct numbers which are publicly known to everyone. And at least t participants can easily reconstruct these secrets by pooling their shadows. The dealer will do the following steps to execute the secret distribution:

- (1) Randomly choose y_1, y_2, \dots, y_n as secret shadows of n participants respectively.
- (2) Construct a polynomial $P_k(x)$ of degree $(t-1)$ and $P_k(0) = s_k$.
- (3) Compute $Z_{k,j} = P_k(x_j) (j=1, 2, \dots, n)$, and then compute $d_{k,j} = Z_{k,j} \oplus f^k(y_j) (j=1, 2, \dots, n)$, where $d_{k,j}$ stands for the shift values and $f^k(y_j)$ the pseudo shadows.
- (4) For $i = k-1, k-2, \dots, 1$, execute the following steps:
 - a) Construct a polynomial $P_i(x)$ of degree $(t-1)$ and $P_i(0) = s_i$.
 - b) For $j=1, 2, \dots, n$, compute $Z_{i,j} = P_i(x_j)$.
 - c) For $j=1, 2, \dots, n$, compute $d_{i,j} = Z_{i,j} \oplus f^i(y_j) \oplus s_{i+1}$, where $d_{i,j}$ stands for the shift values and $f^i(y_j)$ the pseudo shadows.
- (5) Deliver $y_i (i=1, 2, \dots, n)$ to each participant secretly and publish all $d_{i,j}$ for $i=1, 2, \dots, k$ and $j=1, 2, \dots, n$.

In the procedure of the secret reconstruction, at least t participants must first provide their pseudo shadows $f^k(y_i)$, for $i=1, 2, \dots, t$, to reconstruct the secret s_k through the following formula:

$$s_k = P_k(0) = \sum_{a=1}^t (f^k(y_a) \oplus d_{k,a}) \prod_{b=1, b \neq a}^t \frac{0 - x_b}{x_a - x_b}. \tag{1}$$

Then they have to provide their other pseudo shadows in the following special order: $f^{k-1}(y_i), f^{k-2}(y_i), \dots, f^1(y_i)$ for $i=1, 2, \dots, t$, to reconstruct the remaining secrets via the following formula (for $i = k-1, k-2, \dots, 1$):

$$s_i = P_i(0) = \sum_{a=1}^t (f^i(y_a) \oplus d_{i,a} \oplus s_{i+1}) \prod_{b=1, b \neq a}^t \frac{0 - x_b}{x_a - x_b} \quad i = k-1, k-2, \dots, 1. \tag{2}$$

Thus the remaining secrets are reconstructed in the special order: $s_{k-1}, s_{k-2}, \dots, s_1$.

3 The Proposed Scheme

In this section, we shall propose a new multi-stage (t, n) -threshold secret sharing scheme that is an improvement on Chang *et al.*'s scheme [11]. Our scheme needs fewer public values than Chang *et al.*'s scheme. The proposed scheme concludes four parts: (1) system parameters, (2) secret distribution, (3) secret reconstruction and (4) cheater identification. The details of four parts are as follows:

3.1 System Parameters

The proposed scheme works over a finite field $GF(p)$, where p is a large prime number. Suppose that there is a set of n participants $U=\{U_1, U_2, \dots, U_n\}$ in the system. Let D ($D \notin U$) denotes a dealer. In the whole paper, we assume that the dealer is a trusted third party. The dealer randomly selects n distinct integers, $x_i \in [n-t+2, p-1]$, for $i = 1, 2, \dots, n$, as participants' public identifiers and n random integers, $y_i \in [1, p-1]$, for $i = 1, 2, \dots, n$, (i.e. y_i not necessarily distinct) as participants' secret shadows. There are k secrets s_1, s_2, \dots, s_k to be shared among n participants. Let $f(m)$ be a one-way function.

3.2 Secret Distribution

The dealer executes the following steps to implement the secret distribution:

- (1) Construct an n th degree Lagrange interpolation polynomial $P_k(x)$, which passes through the co-ordinates $(x_i, f^k(y_i))$ for $i=1, 2, \dots, n$ and $P_k(0) = s_k$.
- (2) For $j=1, 2, \dots, n-t+1$, Compute the public value $Z_{k,j} = P_k(j)$.
- (3) For $i= k-1, k-2, \dots, 1$, execute the following steps:
 - a) Construct an n th degree Lagrange interpolation polynomial $P_i(x)$, which passes through the co-ordinates $(x_j, f^i(y_j) \oplus s_{i+1})$ for $j=1, 2, \dots, n$ (where “ \oplus ” denotes the exclusive-or operation) and $P_i(0) = s_i$.
 - b) For $j=1, 2, \dots, n-t+1$, compute public values $Z_{i,j} = P_i(j)$.
- (4) Deliver y_i to each participant U_i ($i=1, 2, \dots, n$) secretly and publish all $Z_{i,j}$ for $i=1, 2, \dots, k$ and $j=1, 2, \dots, n-t+1$ in any authenticated manner such as those in [14], [15].

3.3 Secret Reconstruction

In the secret reconstruction phase, at least t participants cooperate to recover the shared secrets by pooling their pseudo shadows. Suppose that a subset of t participants $A = \{U_1, U_2, \dots, U_t\}$ want to recover the shared secrets. These t participants have to first provide their pseudo shadows $f^k(y_i)$, for $i=1, 2, \dots, t$, to reconstruct the secret s_k . According to the public values $Z_{k,j}$ ($j = 1, 2, \dots, n-t+1$) and pseudo shadows $f^k(y_i)$ ($i=1, 2, \dots, t$), the designed combiner can obtain $(n+1)$ co-ordinates altogether, i.e., $(1, P_k(1)), (2, P_k(2)), \dots, (n-t+1, P_k(n-t+1)), (x_1, f^k(y_1)), (x_2, f^k(y_2)), \dots, (x_t, f^k(y_t))$. Without loss of generality, we use (X_i, Y_i) ($i=1, 2, \dots, n+1$) to denote the $(n+1)$ co-ordinates, respectively. Then the secret s_k can be reconstructed through the following formula:

$$s_k = P_k(0) = \sum_{i=1}^{n+1} Y_i \prod_{j=1, j \neq i}^{n+1} \frac{0 - X_j}{X_i - X_j} . \tag{3}$$

In order to recover the remaining secrets, these t participants have to provide their other pseudo shadows in the following special order: $f^{k-1}(y_i), f^{k-2}(y_i), \dots, f^1(y_i)$ for $i=1, 2, \dots, t$. According to the public values $Z_{i,j}$ ($1 \leq i \leq k-1, j = 1, 2, \dots, n-t+1$) and pseudo shadows $f^i(y_i)$ ($i=1, 2, \dots, t$), the designed combiner can obtain $(n+1)$ co-ordinates altogether, i.e., $(1, P_i(1)), (2, P_i(2)), \dots, (n-t+1, P_i(n-t+1)), (x_1, f^i(y_1) \oplus s_{i+1})$,

$(x_2, f^i(y_2) \oplus s_{i+1}), \dots, (x_t, f^i(y_t) \oplus s_{i+1})$. Without loss of generality, we use (X_i, Y_i) ($i=1, 2, \dots, n+1$) to denote the $(n+1)$ co-ordinates, respectively. Then the remaining secrets can be reconstructed via the following formula (for $i = k-1, k-2, \dots, 1$):

$$s_i = P_i(0) = \sum_{j=1}^{n+1} (Y_j \oplus s_{i+1}) \prod_{l=1, l \neq j}^{n+1} \frac{0 - X_l}{X_j - X_l} \quad i = k-1, k-2, \dots, 1. \tag{4}$$

Thus the remaining secrets are reconstructed in the special order: $s_{k-1}, s_{k-2}, \dots, s_1$.

3.4 Cheater Identification

It is very important for any group-oriented secret sharing scheme to detect cheating and identify the cheater, and there are already lots of works on this issue. These methods can be employed in the proposed scheme directly. Readers may refer to works in [16], [17] and [18]. Therefore, we will not iterate on this issue in our paper. But, in order to keep participants' real shadows secret after the cheater identification process, there are some points should be pointed out: the secret dealer should use the pseudo shadow $f^j(y_i)$ of a participant U_i instead of the real shadow y_i of U_i in generating the public values for the cheater identification. At the same time, the verifier should be able to use the pseudo shadow $f^j(y_i)$ to determine whether there exists a cheater without knowing the real shadow y_i . Hence, each participant U_i just pools his/her pseudo shadow $f^j(y_i)$ in the cheater identification process, so that his/her real shadow y_i will not be disclosed by the properties of the one-way function.

4 Analysis and Discussion

In this section, we will prove that our scheme is a real multi-use scheme and that the secrets are constructed stage-by-stage, and then, we shall analyze the performance and security of our scheme.

4.1 Multi-use Scheme

In order to reconstruct the final secret s_1 , at least t participants must provide their pseudo shadows $f^1(y_i)$, for $i=1, 2, \dots, t$. Because $f^1(y_i) \neq y_i$, the secret shadows y_i ($i=1, 2, \dots, t$) still stay un-disclosed. To share the next secret, the dealer only needs to publish a new value r and uses it together with participants' secret shadows y_i ($i=1, 2, \dots, n$) to compute their k th pseudo shadows, i.e., $f^k(y_i \oplus r)$. So the dealer need not deliver a new secret shadow y_i ($i=1, 2, \dots, n$) to every participant over a secret channel, which qualifies our scheme as a multi-use scheme. On the other hand, to share k secrets in our scheme, each participant U_i needs to keep only one secret shadow y_i .

4.2 Multi-stage Feature

Since a secret s_k can only be solved by Equation (3), at least t participants must provide their pseudo shadows $f^k(y_i)$ for $i=1, 2, \dots, t$, first. If they do not have the secret s_k first, they cannot obtain the next secret s_{k-1} by Equation (4). For this reason, they must

forward the order of $f^k(y_i), f^{k-1}(y_i), f^{k-2}(y_i), \dots, f^1(y_i)$ to reconstruct the secrets. So the secrets certainly need to be reconstructed in the special order: s_k, s_{k-1}, \dots, s_1 , that is to say, without knowing the secret s_i , it is computationally impossible to recover the next secret s_{i-1} .

4.3 Performance Analysis

In this subsection, we shall discuss the performance of our scheme in the following two aspects: public values and computational complexity.

(1) Public Value

The number of public values is one of the important parameters that determine the performance of a scheme, because it affects the storage and communication complexity of the scheme [7], [13]. In Chang *et al.*'s scheme, for each sharing secret, it is required to publish n public values, and so there are a total of kn public values for sharing k secrets. However, in the proposed scheme, for each sharing secret, it is required to publish $(n-t+1)$ public values, and only a total of $k(n-t+1)$ public values are required for sharing k secrets. It is easy to see that $kn = k(n-t+1) + k(t-1) \geq k(n-t+1)$, where $t \geq 1$. The proposed scheme becomes more attractive, especially when the threshold t is very close to the number of participants n . For example, when $t = n$, Chang *et al.*'s scheme needs kn public values to share k secrets, while only a total of k public values are required to share k secrets in the proposed scheme.

(2) Computational Complexity

It is obvious that the most time consuming operations of the proposed scheme are the polynomial evaluation and interpolation, which is the same as Chang *et al.*'s scheme. We have efficient $O(n \log^2 n)$ algorithms [19], [20] for these operations. By using these algorithms, the efficiency of our scheme will be improved greatly. Therefore, the proposed scheme is very efficient and easy to implement.

4.4 Security Analysis

Our scheme is mainly based on the Lagrange interpolating polynomial. At least t or more participants pool their secret shadows and easily reconstruct the secret, but only $(t-1)$ or fewer secret shadows will not be enough. Knowing only $(t-1)$ or fewer secret shadows provides no more information about the secret to an opponent than knowing no pieces. In the following, several possible attacks will be raised and fought against to demonstrate the security of the proposed scheme.

Attack 1: A participant U_i tries to reveal other participant' secret shadow y_j , where $1 \leq j \leq n$ and $j \neq i$.

Analysis of Attack 1: When the last secret has been reconstructed, every participant knows others' pseudo shadows $f(y_j)$ ($i=1, 2, \dots, n$) by pooling or computing $P(x_j) \oplus s_2$. However, no one can obtain the true shadow y_j from $f(y_j)$ under the protection of the one-way function f .

Attack 2: A participant U_i tries to reveal other participant' pseudo shadow $f^l(y_j)$ ($1 \leq j \leq n$ and $j \neq i$) that is not published.

Analysis of Attack 2: In our scheme, the secrets are reconstructed in the order of $s_k, s_{k-1}, \dots, s_2, s_1$. To reconstruct the secret s_k , at least t participants should first pool their pseudo shadows $f^k(y_i)$, for $i=1, 2, \dots, t$. However, no one can via $f^k(y_i)$ to obtain the participant's other pseudo shadow $f^l(y_i)$ ($l < k$). It is protected under the one-way function f .

Attack 3: t participants try to disintegrate the order by the dealer's determination to reconstruct the secrets.

Analysis of Attack 3: From the Equation (4), to reconstruct the secret s_i , they should reconstruct the secret s_{i+1} firstly. Thus, the conspiring attack cannot work in our scheme.

From above discussion, it is easy to see that the proposed scheme is a real multi-stage and multi-use scheme, and needs fewer public values than Chang *et al.*'s scheme, so it is higher in efficiency than Chang *et al.*'s scheme is.

5 Conclusions

We made an improvement on Chang *et al.*'s scheme, and proposed a new multi-stage (t, n) -threshold secret sharing scheme. In the proposed scheme, each participant needs to keep only one secret shadow in sharing multiple secrets without updating each participant's secret shadow, and the proposed scheme is a real multi-use and multi-stage scheme and easy to implement. Our scheme needs fewer public values than Chang *et al.*'s scheme, and the implementation of our scheme becomes more attractive, especially when the threshold t is very close to the number of the participants n . Analyses show that the proposed scheme is a computationally secure and efficient scheme and that it provides great capabilities for many applications.

Acknowledgements

Part of this research was supported by the Chinese National Natural Science foundation No. 50479055 and the National Key 973 Project of China, under contract no. G1999035805.

References

1. Chang, T.-Y., Yang, C.-C., Hwang, M.-S.: Threshold Untraceable Signature for Group communications. IEE Proceedings-Communications 151 (2004) 179 - 184
2. Hwang, M. -S., Lee, C. -C., Chang, T. -Y.: Broadcasting Cryptosystem in Computer Networks Using Geometric Properties of Lines. Journal of Information Science and Engineering 18 (2002) 373 - 378
3. Crescenzo, G. D.: Sharing One Secret vs. Sharing Many Secret. Theoretical Computer Science 295 (2003) 123 - 140
4. Shamir, A.: How to Share a Secret. Communications of the ACM 22 (1979) 612 - 613

5. Blakley, G.: Safeguarding Cryptographic Key. In: Smith, M., Zanca J. T. (eds.): Proc. AFIPS 1979 Natl. Conf.. AFIPS Press, New York (1979) 313 - 317
6. Chien, H. Y., Jan, J. K., Tseng, Y. M.: A Practical (t, n) Multi-secret Sharing Scheme. IEICE Transactions on Fundamentals E83-A (12) (2000) 2762 - 2765
7. Pang, L. J., Wang, Y. M.: A New (t, n) Multi-secret Sharing Scheme Based on Shamir's Secret Sharing. Submitted to Applied Mathematics and Computation. Accepted and In Press, Corrected Proof, Available online 5 November 2004
8. Yang, C. C., Chang, T. Y., Hwang, M. S.: A (t, n) Multi-secret Sharing Scheme. Applied Mathematics and Computation 151 (2004) 483 - 490
9. He, J., Dawson, E.: Multistage Secret Sharing Based on One-way Function. Electronics Letters 30 (1994) 1591 - 1592
10. Harn, L.: Comment: Multistage Secret Sharing Based on One-way Function. Electronics Letters 31 (1995) 262
11. Chang, T.Y., Hwang, M. S., Yang, W.P.: A new Multi-stage Secret Sharing Scheme Using One-way Function. ACM SIGOPS Operating Systems 39 (2005) 48 - 55
12. Jackson, Wen-Ai, Martin, Keith, M., O'Keefe, Christine M.: On Sharing Many Secrets. In: Pieprzyk, J., Safavi-Naini, R. (eds.), Asiacrypt'94. Lecture Notes in Computer Science, Vol. 917. Springer-Verlag, Berlin Heidelberg New York (1995) 42 - 54
13. Crescenzo, G. D. L.: Sharing One Secret vs. Sharing Many Secrets: Tight Bounds on the Average Improvement Ratio. In: Proc. of 11th Annu. ACM-SIAM Symp. On Discrete Algorithms. Society for Industrial and Applied Mathematics, San Francisco, California (2000) 273 - 274
14. ElGamal, T.: A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory IT-31 (1985) 469 - 472
15. Rivest, R. L., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM 21 (1978) 120 - 126
16. Wu, T. C., Wu, T. S.: Cheating Detection and Cheater Identification in Secret Sharing Schemes. IEE Proceedings-Computers and Digital Techniques 142 (1995) 367 - 369
17. Tan, K. J., Zhu, H.W., Gu, S.J.: Cheater Identification in (t, n) Threshold Scheme. Computer Communications 22 (1999) 762 - 765
18. Chang, C. C., Hwang, R. J.: Efficient Cheater Identification Method for Threshold Schemes. IEE Proceedings-Computers and Digital Techniques 144 (1997) 23 - 27
19. Aho, A., Hopcroft, J., Ullman, J.: The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, MA (1974)
20. Knuth D.: The Art of Computer Programming. Vol. 2/Seminumerical Algorithms. Addison-Wesley, Reading, MA (1969)

Information Management in E-Learning System

Liu Jing¹, Zheng Li¹, and Yang Fang²

¹ Computer & Information Management Center, Tsinghua University,
100084 Beijing, China
{lj, zhli}@cic.tsinghua.edu.cn

² Department of Foreign Languages, Tsinghua University,
100084 Beijing, China
yangfang@tsinghua.edu.cn

Abstract. The effective management of learning information is crucial for e-learning systems in the digitally networked age. E-learning, an emerging new approach to learning enables learners to access sufficient learning information in different systems. Thus, an increasingly important as well as problematic issue is to achieve the exchangeability and interoperability of information. This paper introduces an effective method and information model for managing learning information via standard metadata, and then proposes an Open Education Service Architecture to achieve the interoperability between heterogeneous learning systems. To better illustrate this architecture, we provide a standard compliant Learning Contents Management System to organize and classify the learning information in an e-learning system.

1 Introduction

Since the wide use of Internet, information and knowledge booms and people's ways of learning have undertaken a revolutionary change. Electronic learning (e-learning) is increasingly accepted as a popular and economical learning method in education as well as professional development programs. A major characteristic of Internet is that plenty of heterogeneous systems coexist and various learning information and learning systems are distributed everywhere. How to make full use of scattered information available and how to manage the information and implement interoperability between those learning systems become one of the major challenges.

Interoperability can be defined as “the ability of two or more systems or components to exchange information and to use the information that has been exchanged” [1]. At present, e-learning information and systems have been applied in all kinds of education programs. Many educational institutions have developed learning systems and resources. Although Internet makes the information public and widespread, the format and content of information vary and the implementation technologies of platform are also different. Because of the lack of effective management and integration, it is very often the case that learning information can only be used in a specific platform. Also, without a standard of information exchange, it is difficult to share the information.

To satisfy the requirement of interoperability, the learning information (e.g. learning content and learner's information) is standardized for accessibility. The standard-

ized information can be shared by different systems. An open architecture is built so that different systems can be integrated via standard interface specification, and the diverse components of systems with their own functions can interoperate with each other.

Web services provide a standard means of communication among different software applications, running on a variety of platforms and/or frameworks [2]. To solve the problem of interoperability between different learning systems, this paper proposes an approach to standardizing architecture of learning systems. The proposed architecture should be services-oriented and it should support China E-Learning Technology Standards (CELTS). [3]

2 History and Current Status of Chinese E-Learning Information Management

While developing e-learning, many countries have realized that the reusability of learning information and system interoperability has economic and practical significance. Works on e-learning information standardization in China began only a few years ago. In 2000, an e-learning technology standardization project started while China e-learning technology standardization committee (CELTSC) [3] was founded. The aim of the project is to develop e-learning standards to enable sharing of information and interoperability among heterogeneous learning systems. But still there are many independent systems and information resources which can not communicate and share with each other effectively. This causes plenty of unnecessary development. It is not only a waste of manpower and material resources, but also an obstacle to communicate with international e-learning systems.

E-learning standardization could be divided into two levels, the first one is learning information standardization, and the second one is common software components and open architectures standardization. Up to now, we have constituted the first set of Chinese e-learning technology standards (CELTS), which include five categories: general standards, learning environment standards, learning resources standards, learner information standards and learning management standards. This set of standards provides data and information model. Standardized learning contents and learner information could be reused in different learning platform.

In China, learning contents and systems are mainly developed by those nonprofit educational institutions and also small-scale enterprises. The developers are dispersive and weakness of fund and technology. The main issue is that how to unite developers and integrate each productions effectively.

It is thus necessary to define an open architecture and specify its application interface so that software components can be shared among organizations. Learning systems can be modularized and loosely coupled via shareable objects and interfaces. While importing new standard components, those existing systems don't need to be overthrown and the modules of systems could be reused and interoperated.

3 E-Learning Information Standardization

Standardization can enable the reusability and interoperability of e-learning information which consists of information on the contents, learners as well as those on tracking the learning process.

3.1 Learning Content Standardization

For learning content management, one of the basic principles is the separation of structure, content and presentation [4]. For the reusability and interoperability learning contents should be broken down into chunks. From a pedagogical perspective, each chunk might play a specific role within an instructional design methodology. Such chunks are called learning objects, which are the basic components of the courseware [5]. Learning objects are composed of some assets which could be texts, images, sounds and any contents that can be present to learners. To describe learning objects, we could use metadata. Metadata is a kind of data which is used to describe data and it provides a means to describe the information of learning objects. We can find, locate and exchange learning objects via metadata. Standards about metadata of learning objects define a set of attributes on learning objects, such as title, key words, format, size, classification etc. The CELTS-3 (China E-Learning Technology Standards), "Specification for Learning Object Metadata" [6], has been developed and widely adopted in China. Different formats of learning content can be packaged with the addition of metadata information. A standardized package can then be used in any learning systems.

3.2 Learner Information Standardization

Learners are the core of learning activity and they play an important role in e-learning. The purpose of building standardized learner information model is to unify the management of students and courses. In the domain of e-learning, learner information includes some specific attributes besides basic personal information, such as learning goal, interest, relationship etc. And we should also define a model which is used to describe competency. The term "competency" is very extensive, which includes knowledge, task, result etc. The standardized information could be exchanged between different systems via XML binding.

3.3 Learning Process Information Management

Learning process can be broken down into several unattached sub processes which should be arranged in a sequence. For example, the order of disposing homework, handing in homework and correct homework can not be reversed. And the whole process should be tracked and noted down by a system which is called Learning Management System (LMS). The CELTS-20, "Specification for learning management system (LMS)" [7] gives a principle of achieving interoperability between LMS. It includes how to manage learning activity, how to evaluate a course and how to access a course in different LMS. However, the implementation of LMS in e-learning is still in its infant stage.

3.4 The Advantage and Importance of Standardization

For the management of those scattered information and distributed systems, standardization solves the major problem. Fig.1 illustrates the standardization of learning contents and information. Learning contents are first classified and packaged. In a package the contents are put into a special order and we use metadata describe the structure of this package. Learner information is defined in a domain model. And standardization regulates the scope of domain. So the attributes of information are limited, which means the problem space is limited. To manage the numerous and complicated information effectively, using standard metadata is almost the only method. Just like network protocol, standardized contents and information are the common language for learning systems and they are the bases of communication between heterogeneous learning systems.

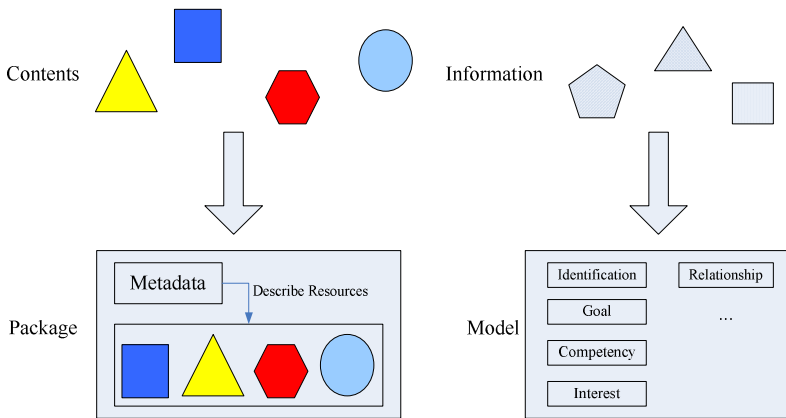


Fig. 1. Standardization of contents and information

4 A Service Oriented E-Learning System Architecture

4.1 The Second Level of Standardization

Learning information standardization is the first level to implement the interoperability between heterogeneous learning systems. Although different learning systems may be implemented in different technologies and may run on different operating systems, their functions of modules are similar such as virtual classroom, chatting room, electronic blackboard, and so on. Such time consuming work have to be repeated in developing every system. To reuse modules in different systems, we must standardize the system architecture. This is the second level of standardization and only standardized information could be retrieved easily in such architecture.

Many international standardization organizations have done some researches in this field. OKI [8] has produced a series of Open Service Interface Definitions (OSIDs) to specify composition of learning systems. All modules designed with this specification can interoperate, which means an existing and well-defined module can

plug and play instead of developing a new module for the same function. Thus the system will become more flexible.

4.2 Web Services and Services-Oriented Architecture

Web Services are a set of standards for describing, publishing, discovering and binding application interfaces which include WSDL (Web Services Description Language), SOAP (Simple Object Access Protocol), UDDI (Universal Description, Discovery and Integration). WSDL is a format for describing a Web Services interface. It is to describe services and a way they should be bound to specific network addresses. The descriptions of services and messages are generally expressed in XML (eXtended Markup Language). SOAP provides the envelope for sending the messages over Internet. UDDI defines a set of services supporting the description and discovery of services providers. Fig.2 illustrates the elements of Web Services.

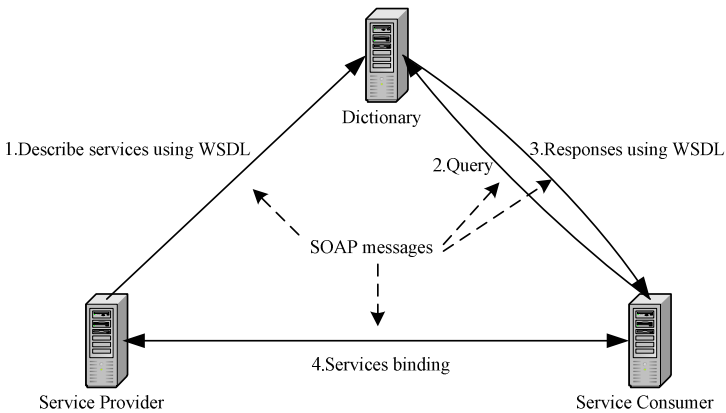


Fig. 2. Elements of web services

SOA (Services-Oriented Architecture) is a kind of methodology in software engineering and it is used to describe distributed resources in heterogeneous network environment. All resources are regarded as services. In essence, SOA is a collection of services. Services can communicate with each other involving simple data transportation and coordinating activity in several services. Although SOA is technology independent, Web Services is still the most suitable implementing technology available for SOA nowadays.

4.3 Why SOA?

Firstly, e-learning is a distributed learning pattern based on Internet and such an environment is what SOA intends to describe. Secondly, to solve the problem of the interoperability between heterogeneous learning systems, a services-oriented design pattern is the most direct and common way. And thirdly, a characteristic of standardization is technology independent, while in a services-based system, services interface is the only place where interaction takes place and the implementation of interface is

transparent for consumers. We define the interface of operation on those standardized information and any implementation compliant to interface specification could be reused in systems. Although the pace of technological development is rapid, such architecture will remain steady within 5 to 10 years' time. In this way, SOA can keep pace with new technology and make the system extensible.

4.4 The Design of E-Learning System Architecture - Open Education Service Architecture

In our e-learning system architecture, modules are grouped into layers from top to bottom. Common function modules can be identified and summarized from existing e-learning systems and real education activities. Then the modules are classified into Educational Services. Between Educational Services Layer and Infrastructure Layer, a Common Services Layer accessing infrastructure, like database and network communication, provides standardized information for upper layer. Thus in Educational Services Layer, interoperation in services level becomes possible. Fig.3 shows layers of Open Education Service Architecture.

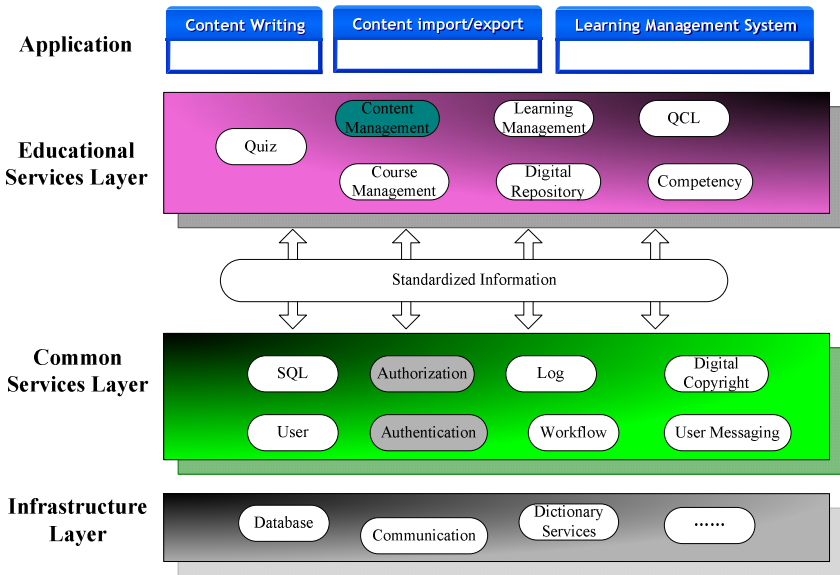


Fig. 3. Layers of open education service architecture

Fig.4 takes an assignment submission as an example and we can see how services work together. When learner submits his assignment, this event as a message will be passed to a logging and tracking service. At the same time, the assignment will be transported to a services broker that is used to manage and schedule services. The broker gets learner's information. If authentication is successful, a submission service will handle the assignment. After the submission is done, its output will return. The result will be noted down by the logging and tracking service, and then the status of learner's assignments will be modified.

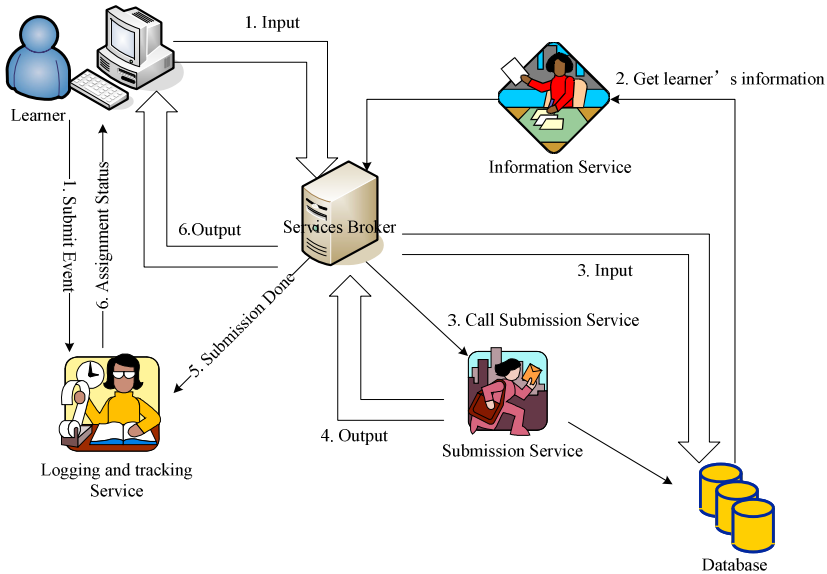


Fig. 4. An example of assignment submission

The learning system is composed of many services with specific function and the learning process can be represented as collaboration of relative services. Services communicate via standardized information whatever their implementations and environments are. This architecture achieves interoperability between heterogeneous learning systems.

4.5 Features of the Architecture

Open. The architecture and interfaces should be open. “Open” doesn’t only mean open source, but also mean “unite” instead of “compete” for merchants. Under such architecture, developers can work in familiar fields and implement their specific interface. Many common and well-developed modules can be reused. Then the application will grow faster and faster.

Loose Coupling and Distributed. Systems are scattered in the architecture and services are distributed. Every module is developed and debugged individually, and then they are integrated into a learning system seamlessly. While the implementation and system environment of services changed, the variation could be unknowable for other services in the system so long as the interface specification is the same.

Re-factory and Flexibility. Systems are built on services and the building structure can be changed, which is called re-factory. This brings us great flexibility for system development and maintenance. Fig.5 is an example of re-factory. Besides the assignment submission application, an assignment check application is required for teachers. This application is composed of four services, three of which are used in assignment submission. Teachers need to download the assignment that students submit, so

a new download service is added. We only develop the new service and build the new application. That means we choose what we need but not develop all of them. And the new assignment check application will not affect original application. This is a basic principle of re-factory – not break down primary function and build new application.

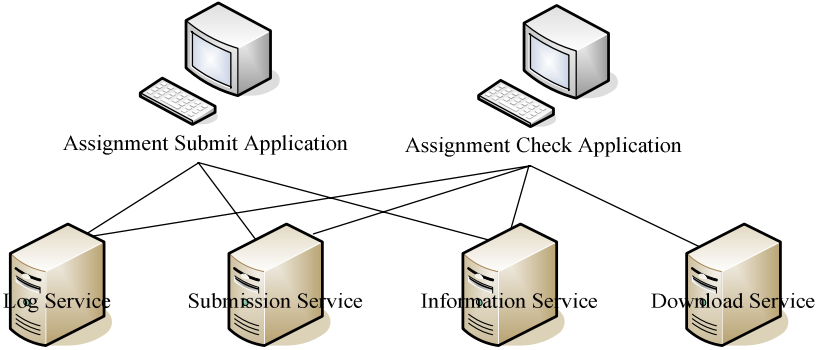


Fig. 5. Re-factory an application from architecture

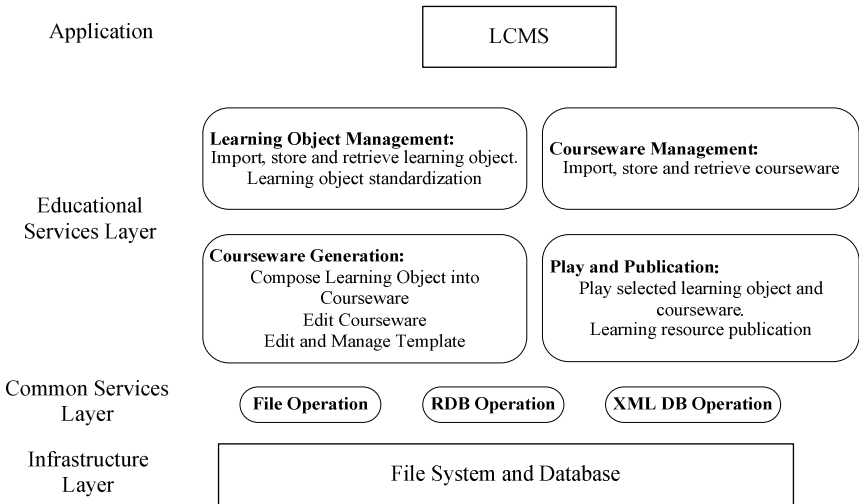


Fig. 6. A Learning Content Management System (LCMS)

Standards Support and Interoperability for Information Exchange. This architecture should support CELTS. Following CELTS, a standardized information model is defined. Learning contents and learner information can be exchanged between various application and platform. Modules in different system can communicate with each other. In another word, the components which were developed in one platform can be used in another platform.

4.6 An Implementation

We have implemented a Learning Content Management System (LCMS) based on the above architecture. According to the architecture we divided the services into two layers: common services and educational services. In common services layer, file operation and database operation (RDB and XML DB) are implemented. And in educational services layer, we have services for learning object management, courseware management, courseware generation and learning resource publication. This Learning Content Management System has been used in our “C++ programming” course, to assist teacher to management learning object such as examples and video courseware. This LCMS can also be an optional additional module for the University e-learning platform “Tsinghua Web School”. In figure 6, the different components and their specific functions will be described.

5 Conclusion and Future Work

To achieve the interoperability between heterogeneous learning systems, standard metadata has been used to describe a uniform content and information model. Further more, the architecture and interface specification of exchanging information need to be defined. Web services technology can satisfy such a requirement. In this paper, we propose a services-oriented Open Education Service Architecture which classifies educational application modules into four layers and supports CELTS.

Further research will work on enriching the architecture with more service and facilitating the application of it with the joint effort of merchants and institutes.

References

1. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries, New York: IEEE, 1990.
2. Xiaofei Liu, El Saddik A., Georganas N.D.: An implementable architecture of an e-learning system. Canadian Conference on Electrical and Computer Engineering, IEEE CCECE 2003, Volume 2, Pages 717-720, May 2003.
3. CELTSC and CELTS. <http://www.celtsc.edu.cn/>
4. Bergstedt S., Wiegrefe S., Wittmann J., Moller D.: Content management systems and e-learning systems -a symbiosis? Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies, ICALT 2003, Pages 155-159, 9-11 July 2003.
5. Wang Xuan, Zheng Li, Yang Fang: An Implementation of Learning Objects Management System. Advances in Web-Based Learning – ICWL 2004 Lecture Notes in Computer Science 3143: 393-399, 2004.
6. China E-learning Technology Standards Committee: Specification for Learning Object Metadata: Information Model, CELTS-3.1 CD 1.6, 2002.
7. China E-learning Technology Standards Committee: Specification for learning management system (LMS), CELTS-20.1 CD 2.0, 2003.
8. Open Knowledge Initiative (OKI). <http://www.okiproject.org/>
9. Web Services and Service-Oriented Architectures. <http://www.service-architecture.com/>
10. Java Technology and Web Services. <http://java.sun.com/webservices/>

A Stratification-Based Approach to Accurate and Fast Image Annotation

Jianye Ye¹, Xiangdong Zhou¹, Jian Pei², Lian Chen¹, and Liang Zhang¹

¹ Department of Computing and Information Technology,
Fudan University Shanghai, China 200433

{042021123, xdzhou, 042021119, zhangl}@fudan.edu.cn

² School of Computing Science, Simon Fraser University, Canada
jpei@cs.sfu.ca

Abstract. Image annotation is an important research problem in content-based image retrieval (CBIR) and computer vision with broad applications. A major challenge is the so-called “semantic gap” between the low-level visual features and the high-level semantic concepts. It is difficult to effectively annotate and extract semantic concepts from an image. In an image with multiple semantic concepts, different objects corresponding to different concepts may often appear in different parts of the image. If we can properly partition the image into regions, it is likely that the semantic concepts are better represented in the regions and thus the annotation of the image as a whole can be more accurate. Motivated by this observation, in this paper we develop a novel *stratification-based* approach to image annotation. First, an image is segmented into some likely meaningful regions. Each region is represented by a set of discretized visual features. A naïve Bayesian method is proposed to model the relationship between the discrete visual features and the semantic concepts. The topic-concept distribution and the significance of the regions in the image are also considered. An extensive experimental study using real data sets shows that our method significantly outperforms many traditional methods. It is comparable to the state-of-the-art Continuous-space Relevance Model in accuracy, but is much more efficient – it is over 200 times faster in our experiments.

1 Introduction

Image annotation is an important research problem in content-based image retrieval (CBIR) and computer vision with broad applications. For a given image, such as a picture, we want to extract the semantics of the image in the form of a list of semantic concepts (or called semantic keywords). For example, the image at the top of Figure 1 can be annotated using three semantic concepts: *tiger*, *stone*, and *grass*.

Typically, automatic annotation can be achieved by supervised learning. A training set of images that are annotated by human experts is provided to train and test an annotation system. After training, the annotation system is to annotate images that are not in the training set. Therefore, the critical problem

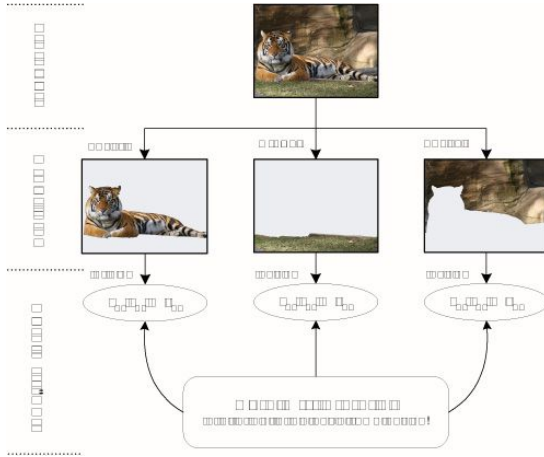


Fig. 1. The stratification Framework

becomes how to efficiently build an accurate model from the training data set and apply the model in the annotation.

Many methods were developed in the previous studies to build effective models for accurate image annotation. Please see Section 4 for a brief review. While the existing methods differ from each other in one way or the other, most of them *treat an image as a set of image blobs* and analyze the semantic concept of each image blob (*that is in the image region level*) to build a vocabulary of *blob word* to represent the whole image.

The essential idea is that an image often has more than one object and thus corresponds to multiple semantic concepts. Treating an image as a whole may not help to identify the features that are highly informative for some specific semantic concepts. For example, in the image shown at the top of Figure 1, the area of yellow/white stripes may strongly suggest the semantic concept of *tiger*, and the area of green with texture may indicate the existence of the semantic concept *grass*. If we can divide the image properly, then we may be able to make a good annotation of the image.

However, classifying image blobs into blob words with respect to correct semantic concepts is still a challenging problem due to the semantic gap. In fact, it is basically another image annotation. Therefore, instead of using blob words to describe the semantic of the image, we believe that it is more effective to represent semantic concepts of images by some low level visual feature descriptors (called “*visual words*”) just analogous to the roles of keywords in text documents¹. Moreover, *visual words* can be learned more accurately at the image blob level than from the whole image.

¹ Recall that the essential building blocks of text documents are keywords instead of paragraphs.

1.1 General Ideas

In this paper, we develop a novel *stratification-based* approach to effective and fast image annotation. The framework is shown in Figure 1.

First, *images are properly segmented into multiple regions so that each region likely corresponds to a single object in the image*. This stratification approach is an application of the divide-and-conquer strategy. The intuition is that fewer objects an area corresponds to, more probably the semantic concepts can be correctly annotated. In order to segment the image properly, we apply the *normalized-cuts method* [13], which partitions an image into regions such that each region is consistent in visual features.

Second, *the visual features of regions should be used to construct the features for annotation*. Naïvely, one may want to extract a semantic concept (i.e., a *blob word*) for each region, and simply take the set of the semantic concepts as the annotation of the whole image. However, such a naïve method may not be effective. The segmentation of images into regions may not be very reliable. Regions are related in an image. That is, directly extracting semantic concepts from a region may not be accurate. Moreover, regions should not be treated equally. For example, a region with a substantially larger area should carry a heavier weight in the determination of the semantics of an image.

Thus, instead of the naïve method, we adopt a more comprehensive approach. We extract the visual features from each region, such as color and texture. Then a learning method is employed to discretize the corresponding visual features of these regions to form a *visual word vocabulary*, analogous to the keyword vocabulary in the text document processing domain. With these visual words, we can describe the image more accurately than using *blob words*.

Last, *a naïve Bayesian method is used to annotate images, and the factors of topic-concept distribution and contributions of regions are also considered*. In the training phase, a naïve Bayesian model is built to capture the correlation between semantic concepts and visual words. Moreover, in many data sets, images are divided according to topics. Two images are in the same topic fold if they share similar semantic concepts. In the annotation phase, we also consider the topic-concept distribution, i.e., the topic information is used in the annotation. We also consider the size of regions – the corresponding visual words are weighted in the annotation phase.

1.2 Our Contributions and the Organization of the Paper

We develop a novel stratification-based approach for image annotation. While the general ideas are described in Section 1.1, the concrete technical approach is developed in the rest of the paper. We conduct an extensive performance study using real data sets and compare with the state-of-the-art methods. The experimental results clearly show that our approach outperforms many traditional image annotation methods, and has an accuracy comparable to the state-of-the-art Continuous-space Relevance Model, but our approach is dramatically more efficient – it is over 200 times faster in our experiments.

The rest of the paper is organized as follows. In Section 2, we discuss how to stratify images and how to extract visual words from regions. Annotation of images using visual words and corresponding algorithms are addressed in Section 3. The factors of topic information and weighted visual words are also considered. We review related work in Section 4. An extensive performance study is reported in Section 5. The paper concludes in Section 6.

2 Stratification of Images and Extraction of Visual Words

2.1 Stratification

Images can be segmented into several sub-regions with particular semantic meanings, analogous to partitioning a text document to paragraphs or sentences. For each region, more lower-level descriptors can be derived such as the visual features, which are analogous to the words in a text document. We can describe the structural organization of images in a hierarchy of three levels, namely images, regions, and visual descriptors, as shown in Figure 1.

According to the idea of stratified image representation, we process each image in the following steps:

1. Image segmentation. We segment each image into different regions using the normalized-cuts method [13], the region is consistent in the visual features.
2. Visual feature extraction for each region, such as color, texture, etc.
3. Image feature discretization and visual word vocabulary generation using a minimal-entropy based method [5].

Given a visual word vocabulary $V = \{v_1, v_2, \dots, v_k\}$, an image I can be represented by a set of segmented regions: $I = \{r_1, \dots, r_m\}$. Each region r_i is represented by a fixed number of visual words: $r_i = \{f_{i1}, \dots, f_{iM}\}$, $f_{ij} \in V$. We define an *indicator function* g on regions and visual words:

$$g(r, f) = \begin{cases} 1 & \text{if region } r \text{ contains visual word } f \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

By summing up visual words from all regions in an image, we represent an image as a vector of visual words: $I = (s_1, \dots, s_k)$, where s_i is the number of regions containing visual word v_i : $s_i = \sum_{j=1}^m g(r_j, v_i)$.

Comparing to previous discrete models which represent an image by a branch of blob words, our stratified image model has two major advantages.

First, generally, the number of blobs in an image is too small after segmentation. For example, in the Corel dataset used by [4,6,7,11], an image only has 1-10 blobs. Nevertheless, an image can have 36-360 visual words in the same dataset, which help our model behave more accurately in probability estimation.

Second, in previous discrete models, a region corresponds to a single blob word after region clustering. All low-level visual feature information are ignored. There is no remedy if a region is clustered incorrectly. Although a similar problem exists in feature discretization, by keeping all low-level visual feature elements we greatly reduce the impact of an incorrect classification.

2.2 Extraction of Visual Words

To generate a visual word vocabulary, we need to convert feature vectors of regions into discrete-valued vectors. Many methods [5,3,9,6] can be used to discretize real-valued data. We employ a supervised method presented by Fayyad and Irani [5] to discrete real-valued visual features. The method is based on a minimal-entropy heuristic. As shown in [3], this method often achieves the best performance among supervised discretization methods.

Minimal-Entropy Based Discretization. Given dataset $S = \{s_1, \dots, s_n\}$ and class label set C , let v_i be the continuous data value of s_i , and c_i be the class label of s_i . The entropy of dataset S is:

$$E(S) = - \sum_{c \in C} \frac{|S(c)|}{|S|} \log \frac{|S(c)|}{|S|}, \tag{2}$$

where $S(c)$ denotes all data points with class label c in S . A boundary T partitions S into two sets $S_1 = \{s_j | v_j < T\}$ and $S_2 = \{s_k | v_k > T\}$. The best boundary T minimizes the entropy after partition, which is:

$$E(T, S) = \frac{|S_1|}{|S|} E(S_1) + \frac{|S_2|}{|S|} E(S_2). \tag{3}$$

Then the boundary selection is repeated on S_1 and S_2 . The partitioning process is applied recursively until some certain stop criteria is satisfied. Information Gain is defined as the entropy reduction after a partition.

$$Gain(T, S) = E(S) - E(T, S). \tag{4}$$

In Fayyad and Irani’s approach [5], the recursive partitioning process stops iff:

$$Gain(T, S) < \frac{\log_2(|S| - 1)}{|S|} + \frac{\Delta(T, S)}{|S|} \tag{5}$$

$$\Delta(T, S) = \log_2(3^k - 2) - [k \cdot E(S) - k_1 \cdot E(S_1) - k_2 \cdot E(S_2)], \tag{6}$$

where k_i denotes the number of class labels represented in S_i . Since partitions are evaluated independently using this criteria during the recursive procedure, the continuous space is not evenly partitioned. Areas having relative high entropy are likely to be partitioned more finely.

Discretization of Visual Features. We describe each region using a 36-dimensional visual feature vector, which includes the average rgb color, the average lab color, area, the mean oriented energy and the area, etc. We assume that each image region inherits all keywords from its parent image. The keywords associated with an image region serve as the class labels for visual features in discretization. Given all data values on one dimension of visual features along with associated class labels, the minimal-entropy based discretization(MED) is

applied to this dimension. Since MED can only handle data with a single label, data with multiple labels needs to be decomposed into multiple data entries each with a single label. For example, we should decompose a data entry valued at 0.35 with class labels “tiger”, “grass”, “sky” into 3 data entries all valued at 0.35 but with class labels “tiger”, “grass”, and “sky”, respectively.

After discretization, each discrete bin can be viewed as a visual word. All discrete bins on all dimensions form the visual word vocabulary. The size of the visual word vocabulary is one of the key aspects influencing the model performance. To control the granularity of discretization, we add a parameter ψ (the value of ψ is 30 in our experiment) to tight the stop criteria:

$$Gain(T, S) < \frac{\log_2(N-1)}{|S|} + \frac{\Delta(T, S)}{\psi \cdot |S|} \quad (7)$$

We also use a minimum partition size to constrain the discretization not to partition too finely in areas which have relative high entropy. After applying this modified MED on every dimension of visual features, we build a vocabulary having 424 visual words.

3 Annotation of Images

3.1 A Naïve Bayesian Model

Given an un-annotated image I , we first employ a segmentation method [13] to split it into regions r_1, \dots, r_n . Then a feature-extraction algorithm [4] is employed to derive a feature vector from a region. Let f_i denote the set of visual words extracted from region r_i , and $f_i = \{a_{i1}, a_{i2}, \dots, a_{im}\}$. For each keyword w in pre-defined keyword vocabulary W , we use the logarithmic probability $\log P(w|f_1, \dots, f_n)$ to estimate the relationship between the image and word w .

$$\log P(w|f_1, \dots, f_n) = \log \frac{P(w) \cdot P(f_1, \dots, f_n|w)}{P(f_1, \dots, f_n)} \quad (8)$$

Without loss of generality, we can assume that $p(w)$ is a constant, and $P(f_1, \dots, f_n)$ is independent of word w , which can be neglected in the computing of the annotation. Thus, we have:

$$\log P(w|f_1, \dots, f_n) \propto \log P(f_1, \dots, f_n|w). \quad (9)$$

The set of visual words from different regions are assumed to be independent.

$$\log P(f_1, \dots, f_n|w) = \sum_{i=1}^n \log P(f_i|w). \quad (10)$$

For each $f_i = (a_{i1}, \dots, a_{im})$, we again treat every visual word in the set statistically independent.

$$\log P(f_i|w) = \sum_{j=1}^m \log P(a_{ij}|w). \quad (11)$$

From (9), (10), and (11), the posterior probability distribution of words for a given un-annotated image I is derived as follows:

$$\log P(w|f_1, \dots, f_n) \propto \sum_{i=1}^n \sum_{j=1}^m \log P(a_{ij}|w) \tag{12}$$

$$P(w|I) = P(w|f_1, \dots, f_n) = \frac{\exp\left(\sum_{i=1}^n \sum_{j=1}^m \log P(a_{ij}|w)\right)}{\sum_{w \in W} \exp\left(\sum_{i=1}^n \sum_{j=1}^m \log P(a_{ij}|w)\right)} \tag{13}$$

The denominator of (13) is a normalizing factor. According to our stratification model, an image is represented by a visual word vector (s_1, \dots, s_k) . Therefore, we can simplify (13):

$$P(w|I) = \frac{\exp\left(\sum_{i=1}^k s_i \log P(v_i|w)\right)}{\sum_{w \in W} \exp\left(\sum_{i=1}^k s_i \log P(v_i|w)\right)} \tag{14}$$

where $\{v_1, v_2, \dots, v_k\}$ denotes the visual word vocabulary. $P(v_i|w)$ can be estimated by counting the occurrence of v_i in all images annotated with w .

3.2 Annotation with Weighted Visual Words

The commonsense tells that *larger the regions, often more decisive to the theme of the image*. Thus, the larger regions should be assigned heavier weights. In this subsection, we discuss an extension to our basic method. In this extended method, the size of regions is considered.

Given image I , Let x_r denote the size proportion of region r to the whole image. The weight of region r is defined according to x_r :

$$w_r = n \cdot \frac{x_r^\alpha}{\sum_{r' \in I} x_{r'}^\alpha}, \tag{15}$$

where n denotes the number of regions in image I and α determines the degree of x_r affecting the weight of visual words in region r . x_r would have more significant influence on region weighting as α increases. We optimize the performance on a small test set to pick the best α value.

We substitute a weight function h for the indicator function g defined in section 3.2:

$$h(r, f) = \begin{cases} w_r & \text{if region } r \text{ contains visual word } f \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

In a weighted stratified image model, we represent an image as a vector of visual words: $I = (s_1, \dots, s_k)$, where s_i is the total weight of regions containing the visual word v_i : $s_i = \sum_{j=1}^n h(r_j, v_i)$.

3.3 Annotation with Topic Information

Most datasets, which can be used as training sets like Corel dataset [4,7,11,6], are categorized into folds. Images within the same fold may have different manual annotation but they all share a similar topic. *Can we use the topic information to improve the annotation?*

For each fold in a training set, a topic model is a word distribution on the annotations of the images in this fold. We estimate the probability of word occurrence by counting the manually annotated words of images in the fold. The heuristics here, which incorporates the topic distribution in the annotation, is that *the keywords with a higher probability in the fold are “popular”, and thus may have a higher probability to be used to label new images exhibiting the similar topic.*

In order to use the topic information, the annotating process for a given un-annotated image is divided into two phases. First, we estimate the probability distribution of the keywords in the un-annotated image by employing equation 13.

Second, given all topic models in the training set, we pick a topic model t which has the most similar word distribution to the un-annotated image. We employ the negative Kullback-Liebler divergence to model the similarity of two distribution [2]. To be more specific, let $P(\cdot|I)$ denote the keyword distribution of un-annotated image I , and TS denote the set of all topic models in the training set. The topic $T(I)$ of image I is defined as follows:

$$T(I) = \arg \max_{T \in TS} \sum_{w \in W} P(w|I) \cdot \log \frac{P(w|T)}{P(w|I)} \quad (17)$$

Then the keyword distribution of I and $T(I)$ are incorporated to determine the final annotation of the image I . Let $G(\cdot|I)$ denote the above mixture distribution, then for each keyword w , we have:

$$G(w|I) = \beta \cdot P(w|I) + (1 - \beta) \cdot P(w|T(I)) \quad (18)$$

where β determines the degree of interpolation between $P(\cdot|I)$ and $P(\cdot|T(I))$. $P(w|T(I))$ can be estimated by counting the occurrence of w in all images in Topic $T(I)$. All the keywords are ranked according to $G(\cdot|I)$, and the top- k keywords with the largest probability are picked out as the annotation of the image.

4 Related Work

To label an image, most previous methods employ segmentation to divide the images into blobs [4,7,11,8], or even grids [12,6], and then the joint probability is estimated on the image regions and the keywords of a pre-defined vocabulary.

As one of the early studies on image annotation, the co-occurrence model proposed by Mori, et al. [12] assigns image annotations to every region of the image. In this model, auto annotation is based on the frequency of the co-occurrence

of a word and an image region. Li and Wang [10] proposed a multi-resolution 2D hidden markov model to view the generation of images as a random process following an underlying probability distribution. The model represents an image by feature vectors under several resolutions. This hierarchical structure helps to catch the spatial context of an image.

More recently, Duygulu, et al. [4] proposed the translation model. In this model, image regions are clustered into a vocabulary of blobs. Each image is represented by a number of blobs from the vocabulary. To annotate images, they employ a classical translation machine model to translate a vocabulary of blobs to a vocabulary of words. Jeon, et al. [7] proposed a cross-media relevance model. Instead of finding the corresponding word for each blob, this model learns the joint probability of all blobs and words within an image. Their experiments showed that the method outperforms the co-occurrence model and translation model significantly on a 5000-image corel dataset. Lavrenko, et al. [11] proposed the continuous-space relevance model which can be viewed as a continuous version of CMRM. They employ a non-parametric kernel-based density estimate to learn the probability of continuous feature vector occurrence. It has a much better performance than the cross-media relevance model but suffers from low efficiency.

Blei and Jordan [1] extended the latent Dirichlet allocation (LDA) model to relate words with images. Feng, Manmatha and Lavrenko [6] proposed the multiple Bernoulli relevance model which uses a Bernoulli process to generate words instead of assuming a multinomial distribution over words. Jin, Cai and Si [8] proposed a coherent language model for image annotation that takes into account the word-word correlation.

5 Empirical Study

5.1 Dataset

In our evaluation, we use a 5000-image corel dataset provided by Duygulu [4]. The data set is available at <http://www.cs.arizona.edu/people/kobus/research/data/eccv2002>. All images are already segmented into regions using normalized-cut algorithm [13]. Each image contains 1-10 image regions. A 36 dimensional feature vector is extracted from each region. There are 371 different concepts used in the manual annotation. Each image is associated with 1-5 words to describe the essential semantics of the image. This dataset was first used by Duygulu to evaluate the translation model [4]. It was also used by Jeon and Lavrenko to evaluate the cross-media relevance model [7] and the continuous-space relevance model [11].

We randomly select 500 images in the data set as the test set and the other 4500 images as the training set.

5.2 Performance Comparison

We compare the annotation performance of our model with other four different models, including the co-occurrence model [12], the translation model [4], the cross-media relevance model [7] and the continuous-space relevance model [11].

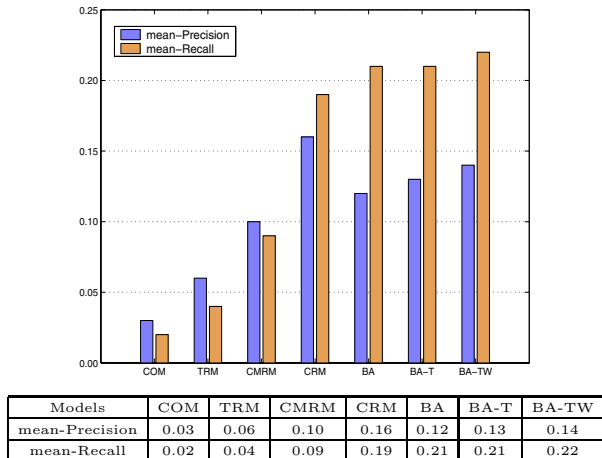


Fig. 2. Mean-Precision and mean-Recall

In our experiments, we first annotate each image in the test set with 5 words. After auto annotation, we use each word from the vocabulary to perform single-word retrieval in the test set. We judge whether an image is correctly retrieved by looking at its manual annotation. We define F measure:

$$F = \frac{2 \cdot \textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}}$$

To examine the contributions of different aspects of our model more clearly, we denote the Bayesian approach without incorporating region weighting and topic model as BA, the Bayesian approach only incorporating topic model as BA-T, the Bayesian approach with region weighting and topic model as BA-TW.

Figures 2 and 3 clearly show that the annotation performance of our model is significantly better than the co-occurrence model, the translation model and the cross-media relevance model. BA-TW has a noticeable improvement over BA and BA-T. The continuous-space relevance model and BA-TW have similar performance in F-measure. Please note that the continuous-space relevance model uses real-valued feature vectors. This is often more effective than the discretize values, but is also much more costly in terms of runtime, as shown in the next subsection.

5.3 Efficiency Comparison

We compare the annotation efficiency of our model with the cross-media relevance model [7] and the continuous-space relevance model [11]. We focus on the efficiency of the three models in annotating new images, so the time cost in training and the time cost in segmentation of un-annotated images are not taken into consideration. We record the time for the three models to annotate

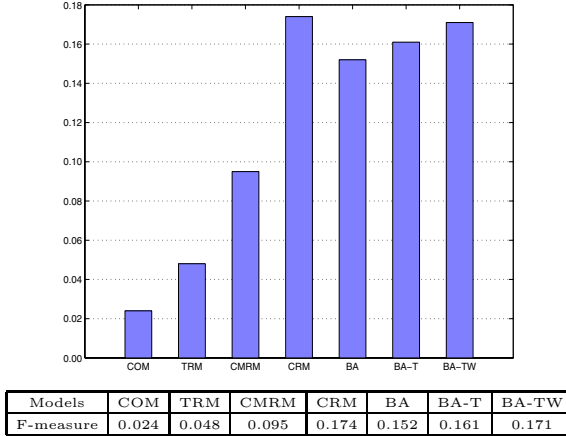


Fig. 3. F-measure in Models

Table 1. Models Efficiency Comparison

Models	CMRM	CRM	BA	BA-T	BA-TW
Time(s)	59.8	4513.1	16.8	21.7	21.8

500 images. All experiments is done on a laptop PC which has one P4 1.8Ghz CPU and 384 Megabytes main memory.

Table 1 shows that our model is about two orders of magnitude faster than CRM and about 3 times faster than CMRM. To understand the experimental results, we analyze the computational complexity of annotating one image in the three models as follows.

From (13), we derive that the computational complexity for our model to annotate one image is $O(|W| \times |V|)$, where W is the text word vocabulary and V is the visual word vocabulary. A large-scale training set is very useful for models to learn the relationships between images and words. Since the computational cost of annotation in CMRM and CRM depends on the size of training set, our model has a better scalability than these two models on large-scale training sets.

6 Conclusions

In this paper, we proposed a stratified image description and a Bayesian model for image annotation. We showed that this model has a good balance between performance and efficiency, as well as a good scalability. Our model employs the Minimal-Entropy based Method for feature discretization. We use the region weighting technique and the topic model-based enhancement to improve the annotation performance.

As the future work, we will explore some semi-naïve Bayesian methods for our model. We believe that a better understanding in the relationship between

regions is very useful for learning the semantics of an image. We will also consider using some multi-dimensional discretization methods to discretize feature vectors in order to catch the dependency of features.

Acknowledgement

This work was supported in part by the NSF of China under grant number 60403018, NSF of Shanghai of China under grant number 04ZR14011, the NSF Grant IIS-0308001, and the NSERC Discovery Grant 312194-05. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Blei, D., Jordan, M.I. (2003): Modeling annotated data. Proc. of the 26th Annual International ACM SIGIR Conference(pp. 127–134). Toronto, Canada: ACM.
2. Croft, W.B. (2000): Combining approaches to information retrieval. In W. B. Croft (Ed.), *Advances in information retrieval*. Cambridge, MA, USA: MIT Press.
3. Dougherty, J., Kohavi, R., Sahami, M. (1995): Supervised and Unsupervised Discretization of Continuous Features. Proc. of the Twelfth International Conference on Machine Learning (pp. 194–202). Tahoe City, California, USA: Morgan Kaufmann.
4. Duygulu, P., Barnard, K., de Freitas, N., Forsyth, D. (2002): Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. Proc. of the Seventh European Conference on Computer Vision (pp. 97–112). Copenhagen, Denmark: Springer.
5. Fayyad, U.M., Irani, K.B. (1993): Multi-interval discretization of continuous-valued attributes for classification learning. Proc. of the 13th International Joint Conference on Artificial Intelligence (pp. 1022–1029). Chambery, France: Morgan Kaufmann.
6. Feng, S.L., Manmatha, R., Lavrenko, V. (2004): Multiple Bernoulli Relevance Models for Image and Video Annotation. IEEE Conference on Computer Vision and Pattern Recognition (pp. 1002–1009). Washington, DC.
7. Jeon, J., Lavrenko, V., Manmatha, R. (2003): Automatic Image Annotation and Retrieval using Cross-Media Relevance Models. Proc. of the 26th Annual International ACM SIGIR Conference (pp. 119–126). Toronto, Canada: ACM.
8. Jin, R., Cai, J.Y., Si, L. (2004): Effective Automatic Image Annotation Via A Coherent Language Model and Active Learning. Proc. of the 12th ACM Annual Conference on Multimedia (ACM MM 2004) New York, USA.
9. Kohavi, R., Sahami, M. (1996): Error-Based and Entropy-Based Discretization of Continuous Features. Proc. of the Second International Conference on Knowledge Discovery and Data Mining (pp. 114–119). Portland, Oregon, USA: AAAI Press.
10. Li, J., Wang, J.Z. (2003): Automatic linguistic indexing of pictures by a statistical modeling approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(9), 1075–1088.
11. Lavrenko, V., Manmatha, R., Jeon, J. (2004): A Model for Learning the Semantics of Pictures. *Advances in Neural Information Processing Systems*. Vancouver, British Columbia, Canada: MIT Press.

12. Mori, Y., Takahashi, H., Oka, R. (1999): Image-to-word transformation based on dividing and vector quantizing images with words. Proc. of the First International Workshop on Multimedia Intelligent Storage and Retrieval Management.
13. Shi, J., Malik, J. (2000): Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9), 1075–1088.
14. Yang, Y., Webb, G.I. (2005): Discretization for data mining. In J. Wang (Ed.), *Encyclopedia of data warehousing and mining*. Idea Group Reference.

Web Services Composition Based on Ontology and Workflow

Huaizhou Yang, Zengzhi Li, Jing Chen, and Hong Xia

Institute of Computer Architecture & networks, Xi'an Jiaotong University,
710049 Xi'an, China
yanghuaizhou@sina.com, lzz@xjtu.edu.cn
jingchen@263.net, Xiahongxiahong@sina.com

Abstract. An approach to compose Web services with ontologies and workflow is presented in this paper, it is illustrated by a travel plan application scenario. Domain-specific ontologies are used to guide and facilitate the interaction among a set of Web services in terms of service utilization scopes with verifiable consistency properties. Through supporting semantic consistency for Web services refinement and reuse, the current technologies are complemented for Web services description, discovery, and composition. Aggregation, specialization, and instantiation are three main operations to support the composition of Web services in the approach. Through the approach, the mechanisms to select Web services according to their utilization scopes are provided, automated means to check if compositions of Web services are semantically correct with respect to these scopes are enabled. The process control logic directed by domain-specific ontologies is combined to our system at both the data level and workflow activity level.¹

1 Introduction

Web services composition is a promising approach to construct next generation large and complex Web applications, in this situation, Web services become the components to form new value-added service. These composite Web applications are built by establishing meaningful data and control flows among individual Web services. These data and control flows form workflows connecting components distributed over the Internet. Traditional workflow management systems are oriented centralized and static process control, which are inadequate for the scalability and versatility requirements of Web applications, such as dynamic restructuring of processes and activities. We need an open environment for developing Web applications using meta data and ontologies to describe data processing patterns. These patterns specify the collection, analysis, and processing of data from a variety of Internet sources, thus providing building blocks for next-generation Semantic Web applications. In this environment, Web services composition is supported by using domain ontologies with object, relation and non-function description. [1]

¹ This work was supported by National Natural Science Foundation of China (No. 90304006) and Research Fund for Doctoral Program of Higher Education of China (No.2002698018).

2 Related Work

The problem set of composing web services with ontology and workflow intersects with many domains including agent-based computing, parallel and distributed computing, semantic Web, workflow, service-oriented architecture(SOA) and some recently developed public web services standards . The related focus is on agent communication, workflow representation and ontologies, service description, discovery and composition.

A set of agent communication language (ACL) standards allow intelligent agents to communicate in an implementation independent manner. Existing ACLs are the Knowledge Query and Manipulation Language (KQML) and the Foundation for Intelligent Physical Agents (FIPA) [2]. A crucial element to this notion of ACLs are ontologies. An ontology provides a shared and common understanding of a domain that can be communicated between people and across application systems. Well-defined ontologies support the automatic processing of information requires a machine readable representation of its semantics. Web Ontology Language (OWL) provide a suitable encoding mechanism for ontological information.

Other research areas directly related to our work are the use of meta data and ontologies for Web services description, discovery and composition, and workflow techniques for scientific processes and Web service composition. Descriptions of the meaning, properties, capabilities, and ontological relationships among Web services, expressed in languages like DAML-S, OWL, support mechanisms to discover, select, activate, compose, and monitor Web resources. These works covers various aspects, ranging from theoretical studies to implementation efforts, from architecture issues to conceptual models [3-5]. Based on these research, complex Web applications can be empowered with semantics and automated reasoning capabilities.

Currently there is a myriad of proposals for specifying Web services composition in intermediate layers, such as WSFL (IBM), BPML (BPMI), XLANG (Microsoft), BPEL4WS (BEA, IBM, Microsoft), WSCI (BEL, Intalio, SAP, Sun), XPDL (WfMC), EDOC (OMG), and UML 2.0 (OMG). These proposals concern the synchronization of the execution of Web services in processes running across enterprise boundaries. They build on top of standards like XML, SOAP, WSDL, and UDDI, providing facilities to interoperate and synchronize the execution of Web services.

3 Application Scenario

3.1 Scenario Description

Travel planning and booking is the most successful business model on the Web, it is also a representative application scenario in service composition research(as Fig.1 shows). Although there are a lot of travel assistant sites on the Web, planning an individual trip on the Web is still a time consuming and complicated venture. Most of the huge number of travel sites provide isolated information about either flights, hotels, rental cars, weather or they relate those information in a very restricted manner. There exists no integrated service for arranging personalized trips to any desired destination,

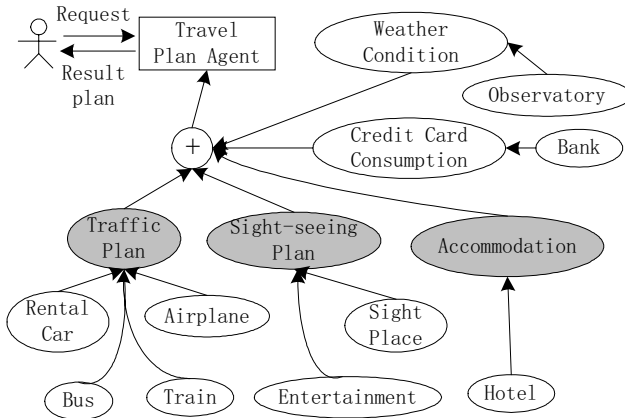


Fig. 1. Travel plan system

relying on distributed information sources which have to be reasonably combined. Recent approaches build on mediators that turn Web sources into structured data sources. Those mediators are the critical component of the whole system because they have to be build individually and kept up to date. What is needed is an individual travel agent which is able to arrange journeys to virtually any place using first hand information from a huge set of different Web sources.

The intelligent agent must obtains relevant data from a variety of heterogeneous information sources (e.g., weather forecast, airplane/train reservation, hotel reservation, sight-seeing ticket booking) to form a individual travel plan based on consumer request. Since gathering and processing real-time data can be costly, database systems and existing documents in different formats are frequently used as alternative sources. Both data and data processing tools can be encapsulated and provided through Web services. These elementary Web services must be described based on domain-specific ontologies to guide and facilitate the interaction among them, further more, form a complex web application system. The top of the system is a workflow management system, the result travel plan is a specific workflow to manage and ensure the plan implementing rightly.

3.2 Technical Challenges

To realize above application scenario, there exist many technical challenges. The outputs of a process can contribute to the inputs of other processes. The data sources to be taken into account and the resulting information. The analysis of the results gives feedback to improve the process or devise new ones. the semantics of data are interrelated with the processes that manipulate them, so that data and processes cannot be completely decoupled. There is no means to track data provenance, i.e., their original source and the way they were obtained and processed. Processes should be documented. When this is done, the specifications produced are broad enough for giving a general view of the processes or formal enough to allow the automatic repetition of the process with different data sets. Lack of catalogs and inspection mechanisms to find and reuse available Web resources to solve each particular problem. Mechanisms

for adaptation and reuse of Web services should be boosted productivity and enhance the quality of the results. Lack of catalogs and inspection mechanisms to find and reuse available Web resources to solve each particular problem. These issues are common to several kinds of applications involving distributed processes over the Web. The following sections describe our approach for handling some of these issues.

4 Semantic Relationships Between Words

There are several semantic relationships between words in the field of linguistics. Let Ω be a set of words referring to objects or concepts from a universe of discourse U . *Objects* are specific instances. *Concepts* are classes that abstractly define and characterize a set of instances or classes. The *universe of discourse* gives a context where the meaning of each word $w \in \Omega$ is stable and consistent. We only consider the following subset in our current research:

- *Synonym*. Two words are *synonyms* of each other if they refer to exactly the same concepts or objects in U .
- *Hypernym/hyponym*. A word w is a *hypernym* of another word w' (conversely w' is a *hyponym* of w) if w refers to a concept that is a generalization of the concept referred to by w' in U . Hyponym is the inverse of hypernym.
- *Holonym/meronym*. A word w is a *holonym* of w' (conversely w' is a *meronym* of w) if w' refers to a concept or object that is part of the one referred to by w in U . Meronym is the inverse of holonym.

Roughly speaking, synonym stands for equivalence of meaning, hypernym for generalization (IS_A), and holonym for aggregation (PART_OF). A set of words Ω is said to be *semantically consistent* for the universe of discourse U and a set of semantic relationships Y if at most one semantic relationship of Y holds between any pair of words in Ω . This ensures some coherence for the meanings of the words in Ω for U .

The semantic relationships defined above preserve certain properties. Let w , w' , and w'' be any three words and θ denote one of the semantic relationships considered. Then, for a given universe of discourse U , the following conditions hold:

- w synonym w (reflexivity)
- $w \theta w' \cap w' \theta w'' \Rightarrow w \theta w''$ (transitivity)
- w synonym $w' \cap w' \theta w'' \Rightarrow w \theta w''$ (transitivity with respect to synonyms)

These properties enable the organization of a set of semantically consistent words Ω according to their semantic relationships in a given universe of discourse U . The *synonym* relationship partitions Ω into a collection of subsets such that the words of each subset are all synonyms. The transitivity of the *hypernym* and *holonym* relationships correlates the semantics of words from different subsets of synonyms, inducing a partial order among the words of Ω . The resulting *arrangement of semantically consistent words* is a directed graph G_Ω that expresses the relative semantics of the words of Ω for the universe of discourse U . The nodes of G_Ω are the subsets of synonyms of Ω . The directed edges of G_Ω represent the semantic relationships among

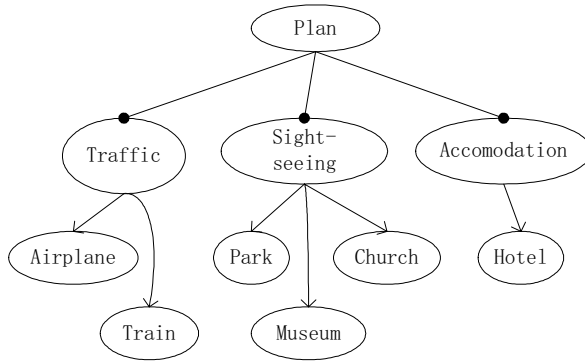


Fig. 2. The Arrangement of concepts

the words of different subsets. As figure 3 illustrated, all the words of Ω represent-concepts, then an arrangement of semantically consistent words is called an arrangement of semantically consistent concepts that appear in the ellipses. The edges representing hypernym relationships are denoted by an arrow close to the specific concept, and the edges representing holonym relationships are denoted by a black circle close to the component concept.

Based on semantic relationships presented above, the meaning of terms used in a particular domain can be described by ontologies. Concretely, ontologies describe the utilization scopes of data sets and processes and orient the refinement and composition of Web services. A utilization scope is a context in which different data sets and distinct versions of a repertoire of services can be used. These are the foundations of our scheme to catalog and reuse components and ensure the semantic consistency of the resulting Web services compositions.

5 The Activity Model and Process

5.1 The Definition of Activity Model

Activity pattern is the basic component to construct a system. It is an abstraction that defines the structure and behavior of a collection of instances of data processing activities implemented as Web services. Each activity pattern is designed to solve a well-defined category of problems in a particular utilization scope. An activity pattern has a set of communication ports, called *parameters*, to exchange data with other activity patterns and data repositories. Each parameter of an activity pattern refers to a Web service encapsulating a data source or sink for that particular pattern. Each input parameter is associated with outputs of another activity pattern or with a data repository. Conversely, each output parameter is associated with inputs of another activity pattern or with a data repository.

Definition 5.1. An activity pattern a is a five-tuple $(NAME, COVER, IN, OUT, TASK)$ where:

NAME is the string used as the name of a

COVER is the ontological coverage of a i.e., expresses its utilization scope

IN is the list of input parameters of α
OUT is the list of output parameters of α
TASK describes the processing chores of α

5.2 Activity Pattern Aggregation

A complex activity pattern is defined as an aggregation of a set of component activity patterns. A component activity pattern can itself be a complex activity pattern or an elementary activity pattern. When decomposing an activity pattern into its constituents, or composing an activity pattern from the components, we have to make sure that there is no conflict among names and ontological coverages of the activity patterns involved and that all parameters are connected.

Definition 5.2. Activity pattern α is an aggregation of the activity patterns β_1, \dots, β_n ($n \geq 1$) if the following conditions are verified (let $1 \leq i, j \leq n$; $i \neq j$ for each condition):

1. $\forall \beta_i : \text{NAME}(\alpha) \neq \text{NAME}(\beta_i) \cup \text{COVER}(\alpha) \neq \text{COVER}(\beta_i)$
2. $\forall \beta_i, \beta_j : \text{NAME}(\beta_i) \neq \text{NAME}(\beta_j) \cup \text{COVER}(\beta_i) \neq \text{COVER}(\beta_j)$
3. $\forall \beta_i : \text{COVER}(\alpha) \supset \text{COVER}(\beta_i) \cup \text{COVER}(\beta_i) \supset \text{COVER}(\alpha)$
4. $\forall p \in \text{IN}(\alpha) : \exists \beta_i$ such that $p \in \text{IN}(\beta_i)$
5. $\forall p \in \text{OUT}(\alpha) : \exists \beta_i$ such that $p \in \text{OUT}(\beta_i)$
6. $\forall \beta_i, p' \in \text{IN}(\beta_i) : p' \in \text{IN}(\alpha) \cup (\exists \beta_j$ such that $p' \in \text{OUT}(\beta_j))$
7. $\forall \beta_i, p' \in \text{OUT}(\beta_i) : p' \in \text{OUT}(\alpha) \cup (\exists \beta_j$ such that $p' \in \text{IN}(\beta_j))$

Definition 5.2 states that an activity pattern α is defined as an aggregation of n component activity patterns β_1, \dots, β_n if they satisfy the above-mentioned seven conditions. Condition 1 says that the name and the ontological coverage of each constituent pattern β_i must be different from the name and coverage of the aggregated activity pattern. Condition 2 specifies that the name and coverage of a constituent activity pattern can uniquely distinguish itself from other constituent patterns of α . Condition 3 states that the ontological coverage of the composite pattern α must encompass the coverage of each constituent pattern β_i or vice versa, i.e., the intersection of their utilization scopes is not null. Condition 4 and 5 ensures that every input(or output) parameter of α is connected to an input(or output) parameter of some constituent β_i . Finally, conditions 6 and 7 state that all parameters of constituent patterns must be connected to a parameter of other constituent or the aggregated pattern.

5.3 Activity Pattern Specialization

The descriptors of an activity pattern can be refined when specializing that activity pattern for a particular situation.

Definition 5.3. Activity pattern β is a specialization of the activity pattern α (conversely α is a generalization of β) if the following conditions are verified:

1. $\text{NAME}(\alpha) \neq \text{NAME}(\beta) \cup \text{COVER}(\alpha) \neq \text{COVER}(\beta)$
2. $\text{COVER}(\alpha) \supset \text{COVER}(\beta)$
3. $\forall p \in \text{IN}(\alpha) : \exists p' \in \text{IN}(\beta)$ such that $p \mapsto p'$
4. $\forall p \in \text{OUT}(\alpha) : \exists p' \in \text{OUT}(\beta)$ such that $p \mapsto p'$

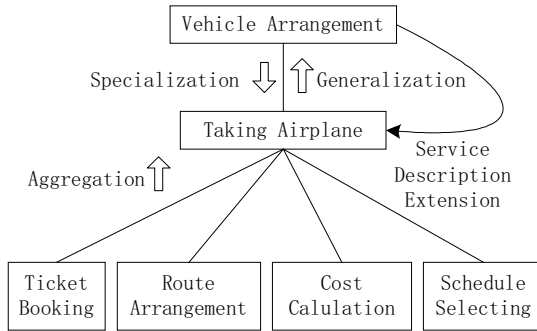


Fig. 3. Combining specialization and aggregation

We call α the generalized activity pattern of β and β a specialized activity pattern (version) of α . Condition 1 of definition 5.3 states that the name and/or ontological coverage of the generalized activity pattern α must be different from those of its specialized version β . Condition 2 states that the ontological coverage of α must encompass that of β . The notation $p \mapsto p'$ in conditions 3 and 4 means that each parameter p' of β must refer to a Web service that is a refinement of the Web service referred to by the corresponding parameter p of α . This refinement of Web services can refer to their capabilities or data contents. The exact relationship between the generic and the refined parameters is defined in the description of the corresponding Web services. Ontological coverages can be associated with these Web services to express and correlate their utilization scopes.

The aggregation and specialization of activity patterns can be combined to define a complex activity pattern whose constituents depend on the utilization scope to which the complex pattern is specialized. The definition of such a complex activity pattern must conform to both the conditions of aggregation and the conditions of specialization.

5.4 Process Framework and Process Specification

Tuples of terms taken from domain specific ontologies, called ontological coverages, formally describe and organize the utilization scopes of Web services. A hierarchy of activity patterns is called a *process framework*.

Definition 5.4. A *process framework* is a directed graph $\Phi(V_\Phi, E_\Phi)$ satisfying the following conditions:

1. V_Φ is the set of vertices of Φ
2. E_Φ is the set of edges of Φ
3. $\forall v \in V_\Phi: v$ is an activity pattern
4. $(v, v') \in E_\Phi \Leftrightarrow v' \text{ constituent } v \cup v' \text{ specialization } v$
5. Φ is acyclic
6. Φ is connected

Definition 5.4 establishes the structural properties of a process framework – a directed graph $\Phi(V_\Phi, E_\Phi)$ whose nodes represent the activity patterns and whose directed edges correspond to the aggregation and specialization relationships among

these patterns. Condition 4 states that there is a directed edge (v, v') from vertex v to vertex v' in Φ if and only if v' is a constituent of v or v' is a specialization of v . Condition 5 states that no sequence of aggregations and/or specializations of patterns in Φ can lead from one pattern to itself. This restriction is necessary because aggregation and specialization can intermingle. In such a case, an aggregation may break the gradual narrowing of the utilization scopes achieved by specialization. Condition 6 guarantees the connectivity of the activity patterns participating in the process framework Φ .

Each activity pattern of a process framework is associated with an ontological coverage that expresses its utilization scope in order to drive the selection and reuse of components. A process framework must be refined, adapted to a particular situation, and instantiated before execution. Some rules to check the semantic consistency of process frameworks and instantiated processes based on correlations of the ontological coverages of their constituents can be provided. For example, the ontological coverages of all the components of a process framework must be compatible with (encompass or be encompassed by) the ontological coverage of the highest activity in the hierarchy. Aggregation, specialization, and instantiation of activity patterns are employed to organize and reuse the components of processes. A process framework captures the possibilities for reusing and composing Web services to build consistent processes for different situations in terms of utilization scopes, data dependencies, and execution dependencies among components. The adaptation of a process framework for a particular scope consists in choosing components to compose a process tailored for that scope.

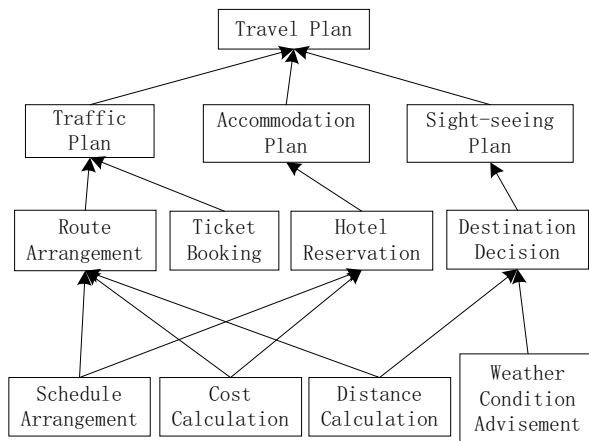


Fig. 4. Process framework for travel plan

Definition 5.5. A process specification $\Pi(V_{\Pi}, E_{\Pi})$ associated with a utilization scope expressed by an ontological coverage C is a subgraph of a process framework satisfying the properties:

1. $\forall (v, v') \in E_{\Pi} : v' \text{ constituent } v$
2. $\forall v \in V_{\Pi} : (\exists v' \in V_{\Pi} \text{ such that } (v, v') \in E_{\Pi}) \Rightarrow v \text{ is atomic}$
3. $\forall v \in V_{\Pi} : COVER(v) \supset C$

In Definition 5.5, condition 1 states that Π is a decomposition hierarchy, i.e., all its edges refer to aggregations of activity patterns. Condition 2 states that all the leaves of Π are atomic patterns, otherwise Π would be missing some constituents for its execution. Condition 3 ensures that the ontological coverage of each pattern participating in Π encompasses the coverage C associated with Π , i.e., the intersection of the utilization scopes of all the constituents of Π are equivalent or contain the utilization scope of Π . Frameworks, specific processes, or individual activity patterns can always be reused to produce new or extended frameworks. Additionally, when adapting a framework, the development of activity patterns to contemplate specific needs also contributes to enrich the repertoire of specialized patterns of a framework.

6 Implementation

All the elements presented above are at the conceptual level. Thus, after adapting a process framework to produce a process specification for a particular situation, this process has to be instantiated for execution. Instantiating a process specification consists in assigning concrete Web services to handle the inputs and outputs of each activity pattern, allocating sites to execute the corresponding tasks and designating agents to perform them. Once particular resources have been assigned, the specific formats and protocols used to connect them can be defined. This may be done by using the binding mechanisms of Web services specification languages like WSDL.

6.1 System Architecture

We implemented a workflow engine that reads and executes an ontology-based web service workflow instance that is a process specification of our process framework. The core modules are workflow management, ontology management, and Web service agent. Workflow management module is responsible for loading, creating, saving and retrieving workflow instances. Ontology management module implements reading, writing, updating and validating a workflow instance based on ontology information of constituent Web services. Workflow engine module holds the workflow execution logic. Web service agent module contains an agent that uses a workflow instance to send messages to web services as instructed by the workflow engine. WS agent module uses registry server based on Universal Description, Discovery and Integration (UDDI) to find the web services. The implementation uses WebSphere Studio Application Developer (WSAD) and IBM ETTK-WS for web services development. Jena toolkit from HP Labs is used for OWL ontology reading and writing. In our experience, the workflow instance specifications using OWL ontology offers the following benefits:

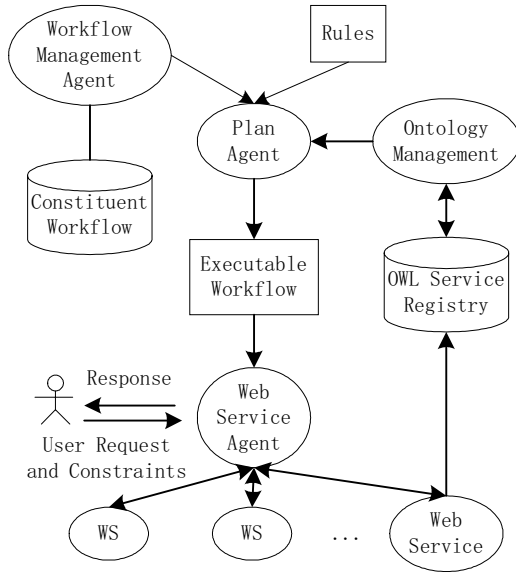


Fig. 5. The architecture of travel plan system

- Activity pattern and concepts can be ordered freely.
- Many constraints can be made explicit in the ontology, instead of being in parser code.
- Ontology coverage and their relationship can be described explicitly and freely.
- Class-based specification is easy to map to implementation classes.

Our main motivation for developing an workflow engine based on ontology was to enable constituent Web services sharing and composition using automated reasoning.

6.2 Travel Plan Agent

Agent here is an autonomous software entity that acts on behalf of the user. The role of travel plan agent is to plan a workflow that accomplishes the task user needs. The workflow is created at run-time, based on following:

- Shared registry information about web services
- Shared workflow instances for component web services
- Main workflow instance
- User input parameters
- Rules

The main workflow instance is described using BPEL. The constituent Web services are described on OWL. A separate workflow management agent is proposed separately, it stores explicitly registered workflow instances in the same sense as UDDI registry stores web service interface descriptions. These instances enables plan agent to select services based on their workflow and ontology coverage. The plan

agent can automatically make the composition based on the stored workflows. Figure 5 presents the inputs and output of the plan agent. Plan agent implementation will form a knowledge base from service ontology description and workflow instances, further more to enable making logic queries [6]. The implementation is not yet able to produce an executable workflow instance for our workflow engine. One specific challenge is to select a single workflow instance from the several possibilities.

7 Conclusion

To compose Web services with ontologies and workflow is still in infancy phase. Web Services are standards that enable remote invocation among heterogeneous systems and hence a perfect solution to leverage diverse legacy systems as well. Ontology releases the possibility of machine readability and of precise understanding among parties. In this paper, through modeling the semantic relationships among specific domain Web services, formally describing the constituent operations, i.e., activity pattern aggregation, activity pattern specialization and process specification, we construct the basic Web service composition infrastructure based on agents and workflow. However, many important issues must be addressed in our further work, such as the extensive service functions and QoS description, the policies of service discovery and selection, the choreography of Web services into e-commerce transactions, the security issues in the platform, the dynamic service composition powered by automated reasoning. Only when these issues had been settled, a entire solution would have been reached, the new-style Web service composition platform could be implemented.

References

1. Andrew W, Anand P, Brian B. Local consensus ontologies for B2B-oriented service composition[A]. AAMAS'03[C]. Melbourne, Australia, 2003, 7. 647-654.
2. Nolan J, Sood A, Simon R. Discovering and composing distributed atomic agents for imagery and geospatial problem solving[J]. International Journal of Pattern Recognition and Artificial Intelligence, 2002, 16(8):995-1019.
3. Brahim M, Athman B, Ahmed K. Composing Web services on the semantic Web[J]. The International Journal on Very Large Data Bases, 2003, 10 (4):333-351.
4. Tse-Ming T, Han-Kuan Y, Ping-Yao L, et al. Semantic Modeling among Web services interfaces for services integration-SOTA (smart office task automation) platform[A]. DEXA'03[C]. Prague, Czech Republic, 2003, 9. 579-583.
5. Declan O, David L. Semantically driven service interoperability for pervasive computing[A]. MobiDE'03 [C]. San Diego, USA, 2003, 9. 17-24.
6. Christoph B. Semantic Web services: reflections on Web service mediation and composition[A]. WISE'03[C]. Roma, Italy, 2003, 10. 253-260.

Web Service Composition Using Markov Decision Processes

Aiqiang Gao, Dongqing Yang, Shiwei Tang, and Ming Zhang

School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, China
{aqqao, ydq, mzhang}@db.pku.edu.cn
tsw@pku.edu.cn

Abstract. This paper discusses a method for dynamic web service composition, which is based on Markov Decision Processes (MDP). It is defined on the base of QoS description and addresses the issue of selecting web services for the purpose of their composition. Web service composition patterns including sequential, conditional, parallel and iterative are modeled in MDP. Two computation algorithms for MDP are introduced and implemented. One is backward recursive iteration and the other is forward iteration, both called Value Iteration. Experiments are conducted to evaluate the performance of MDP methods for service composition. Experimental results show that the methods are efficient. The effect of average failure probability of service invocation on performance is also examined in this paper.

1 Introduction

While the Web is oriented to information sharing and human-machine interaction, the emerging paradigm of web services promise to bring to distributed computation and services the flexibility that the web has brought to the sharing of documents (see [1]).

Web service composition ([1, 2]) provides a way to combine basic web services (possibly offered by different providers) into value-added services that satisfy user needs. The same class of web services for a task may be provided by one or more providers, any one of which can be invoked to execute the targeted task. Web services provided by different providers may be different in Quality of Service (QoS), so decision needs to be made to determine which services are to participate in a given composite one.

Because web service requests are inherently stochastic(see [3]) and web services execution environment is not deterministic, it is inappropriate to select and compose web services statically. In this paper, Markov Decision Processes are used to model the process of dynamic web services selection and composition. It is defined on the base of QoS description for web services. The process produces a policy that could optimally guide the composite web service towards its goal, which is based on Bellman's Principle of Optimality(see [4]).

Two computation algorithms for MDP are introduced and implemented. The first one is backward recursive value iteration. Because this approach can not handle the cases when web service invocation failure occurs, value iteration starting from beginning of the decision process is introduced. With forward iteration, the process of policy computation is interwoven with the process of service execution tracking, which re-evaluates the QoS data for invoked web services. It is assumed that the execution history of services is recorded and QoS data about web services can be accessed in service registries. Experiments are conducted to evaluate the performance of MDP methods for service composition. The results show the methods are efficient.

This paper is organized as follows: Section 2 describes web service model, for both elementary services and composite ones; Section 3 gives Markov Decision Process method for composite web service; Section 4 introduces two algorithms for computing MDP; Section 5 shows experimental results of the approaches introduced in section 4; Section 6 reviews related works and Section 7 concludes this paper and discusses future work.

2 Web Service Model

In this paper, web services of the same kind are aggregated to be a service class (or called abstract service). The interfaces of web services that belong to a service class are identical, though they may be different in quality of service.

2.1 Model for Single Web Service

For an application to use a web service, its programmatic interfaces must be precisely described. WSDL [9] is an XML grammar for specifying properties of a web service such as *what* it does, *where* it is located and *how* it is invoked.

Though the interfaces are essential to automatic composition and verification of composite services, QoS properties of web service provide supports for dynamic service selection and composition. QoS model for Web service in this paper includes four criteria: reliability, cost, response time and availability.

Reliability (successful invocation rate): Successful invocation rate is the probability that a service request is responded successfully. Its value is computed using the expression $Num(success)/Num(all)$, where $Num(success)$ records the number of successful invocation and $Num(all)$ the number of all invocation.

Cost: The cost $cost(s, op)$ is the cost that a service requester has to pay for invoking the operation op .

Response time: $response(s)$ measures the expected delay between the moment when a request is sent and the moment when the results are received.

Availability: The availability $availability(s)$ of a service s is the probability that the service is accessible.

So, the quality vector for an operation op of a service s is defined as

$$\text{quality}(s, op) = (cost(s, op), response(s, op), reliability(s), availability(s))$$

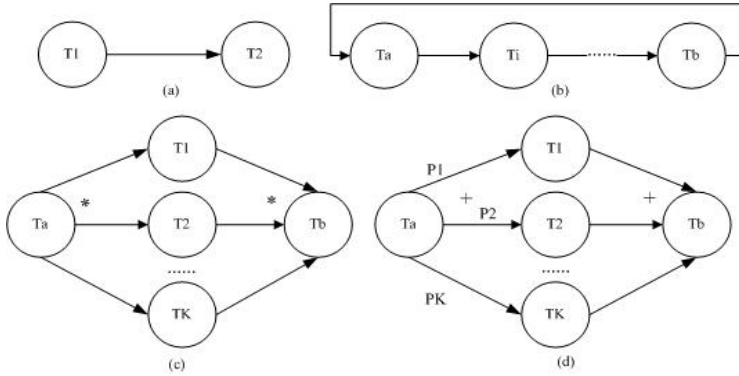


Fig. 1. Composite service constructs. (a) sequential; (b) iteration; (c) parallel denoted as symbol *; (d) conditional, with probability p_1, p_2, \dots, p_K , denoted as +.

2.2 Web Service Composition

The standards for web service composition have been proposed for years, including BPEL4WS[6], WS-Choreography[7], OWL-S(DAML-S)[8] and so on. The constructs and composition patterns in those standards or languages can be summarized as workflow patterns discussed by [10] in detail. The usually used patterns are sequential, conditional choice (or called switch), parallel and iterative. This paper will only discuss how to represent these four constructs in Markov decision process.

Composite web services are defined as abstract representation of business process using service class. The task nodes are defined independent of any concrete web services, which means any concrete service that satisfies the interfaces description can be bound to the node. Fig. 1 gives the illustration of some web service composition scenarios.

3 Markov Decision Processes for Web Service Composition

In this section, the concepts for Markov Decision Process are first introduced, which are taken from [4, 13, 14]. Then MDP for web service composition is defined in the following two subsections.

3.1 Preliminary Concepts

Definition 1. (Markov Decision Process (MDP))

A Markov Decision Process is a 5-tuple, $\{T, S, A(i), p(\cdot|i, a), r(i, a)\}$, where T is the set of all decision time units.

For each value of time unit $t \in \{0, 1, \dots, N\}$, system state is denoted as i . The set of all actions that may be taken at time t is denoted as $A(i)$ which is also called action spaces. The union of all $A(i)$ is denoted as $A = \bigcup_{i \in S} A(i)$. It is assumed that state and action sets are finite and independent of time.

At any time unit t , two results will be achieved after action $a \in A(i)$ is taken: (1) a reward $r(i, a)$ (2) system state at next time unit is determined by probability distribution $p(\cdot|i, a)$

$R(i, a)$ is a real function defined on $i \in S$ and $a \in A(i)$. It represents reward when $r(i, a)$ is positive, cost or penalties while negative. Generally speaking, the reward depends on state j at next time unit, i.e., $r(i, a, j)$. The expected reward of action a is $r(i, a) = \sum_{j \in S} r(i, a, j)p(j|i, a)$.

Function $p(j|i, a)$ is transition probability function which denotes the probability that the system will transfer to state j from state i . It is assumed that the sum of the probability of all states the system will transfer to from some state i is 1. That is to say: $\sum_{j \in S} p(j|i, a) = 1$.

Markov Decision Processes are special cases of decision processes where transition probability and reward function depend only on current state and action taken, independent of history.

Definition 2. (Decision rule and Policy)

A decision rule δ_t , for time t , determines a probability distribution of actions over $A(i)$. It is the principle that guides taking action at any state.

A policy is a sequence of decision rules. A policy is denoted by π and takes the form $\pi = (\delta_1, \delta_2, \dots, \delta_t, \dots)$. Symbol π is used for both infinite and finite horizon problems. To obtain a finite horizon policy from an infinite horizon policy, its operation must be restricted to a finite number of time units. A policy tells how to determine actions for any time unit of the process.

The decision problem is to compute a policy that will maximize the value of some optimality criteria before the decision process begins. The criteria of Markov decision process used in this paper is finite horizon total expected reward.

3.2 MDP for Web Service Composition

Markov Decision Processes for web service composition are defined in the follows way.

Decision Time Unit

Because the number of nodes defined in a composite web service is usually finite, finite horizon MDP is used to represent the composition problem.

States

State is defined as a conjunction of status of each task node. Suppose there are M task nodes, then a system state of composite service is written as a M -tuple: $\langle s_1 \dots s_i \dots s_M \rangle (1 \leq i \leq M, 0 \leq s_i \leq 1)$, where $s_i = 1$ represents that this node is active and has been bound to a concrete web service while $s_i = 0$ means that this node is not active.

Action

Actions may be take n at some state is the set of candidate web services that can be bound to the current active task node. For node i if there are M_i candidate services, then the cardinality of the set $A(i)$ is M_i .

Transition Probability Matrix

The state where node i is active is written as $\langle s_1 \dots s_i \dots \rangle$ with $s_i = 1$, from which system will enter state $\langle s_1 \dots s_i \dots s_j \dots \rangle$ (with $s_j = 1$) with probability of $reliability(s_{i,j})$ or stay at state $\langle s_1 \dots s_i \dots \rangle$ with probability of $1 - reliability(s_{i,j})$.

Invocation of service $s_{i,j}$ is the result of decision rule computed by value iteration approach for MDP. If the invocation of web service $s_{i,j}$ fails, state remains unchanged and the transition probability will be adjusted accordingly. After that, policy computation for MDP is resumed and new action might be picked up according to adjusted transition probability matrix.

Reward

Reward is defined as response time or cost associated with service $s_{i,j}$ that is selected to bind to node i . The goal of web service composition problem is usually to find the solution which minimizes response time. According to the definition of MDP, response time is negative reward, which means the policy is to minimize the total expected response time.

3.3 Discussion for Composition Constructs

Definition of MDP for web service composition in last subsection is for sequential construct. Representation of conditional, parallel and iterative constructs using MDP (as shown in Fig. 1) will be discussed in this subsection.

Conditional Case

Suppose there are K choices in a conditional construct and the probability associated with each branch is p_1, \dots, p_K , respectively.

If current state is $\langle s_1 \dots s_i \dots s_M \rangle$ with s_i corresponding to entry task t_a , system will enter state $\langle s_1 \dots s_i \dots s_{j_1} \dots s_M \rangle \dots \langle s_1 \dots s_i \dots s_{j_K} \dots s_M \rangle$ with probability $p_1 * reliability(s_{i,j_1}) \dots p_K * reliability(s_{i,j_K})$, respectively.

State $S(jk)$ ($k \in \{1, 2, \dots, K\}$) corresponds to the state where node jk is active, which denotes the k -th branch of the conditional construct.

$Reliability(s_{i,jk})(k \in \{1, 2, \dots, K\})$ denotes the probability that the system will transfer from state $S(i)$ to state $S(jk)$, where $s_{i,jk}$ is the web service selected for execution of node i .

The reward is defined as the corresponding response time of the selected web service.

For task t_b enclosing the conditional construct, the definition of transition probability is in similar way. Suppose the k -th branch is entered from task t_a , then system will enter state $\langle s_1 \dots s_i \dots s_{t_b} \dots s_M \rangle$ from state $S(jk)$ with probability $reliability(s_{jk,t_b})$.

Parallel Case

Suppose current state is $\langle s_1 \dots s_i \dots s_j \dots s_M \rangle$ with s_i corresponding to entry task t_a . Then system will enter state $\langle s_1 \dots s_{j_1} \dots s_{j_K} \dots s_M \rangle$ as a result of a parallel construct, where $s_{j_1} \dots s_{j_K}$ are all active. The corresponding transition probability is $\prod_{1 \leq k \leq K} Reliability(s_{i,j_k})$, where s_{i,j_k} is the web service picked for executing node i . Reward is defined as the maximum of all these K web services.

Iterative Case

To represent iterative construct in MDP, it is not necessary to unfold it into a sequential construct as done by [11].

Suppose t_a and t_b is the entry and exit of an iterative construct, respectively. Tasks within the loop can be modeled using sequential, conditional and parallel ones. Only the exit task needs special discussion. After task t_b finishes, the probability that system will break out the loop is p , which means system will transfer from state for t_b to states for tasks following the iterative with probability p . And system will remain at state for t_b with probability $1 - p$.

4 Algorithms for Markov Decision Processes

4.1 Backward Recursive Value Iteration

MDP can be computed by backward recursive value iteration. It starts from horizon N given final reward $r_N(i_N)$ and iterates to horizon 0. When horizon 0 is reached, optimal criteria and optimal actions at each horizon are computed. The sequence of actions computed from this algorithm form the policy. Algorithm 1 outlines the computation procedure. More details about this algorithm can be found in [4, 13, 14].

Algorithm 1. Value Iteration

1: Step1: let $t := N$ and $\forall i_N \in S$,

$$u_N^*(i_N) = r_N(i_N) \tag{1}$$

2: Step2:

3: **if** $t = 0$ **then**

4: $\pi = (f_0^*, f_1^*, \dots, f_{N-1}^*)$ is the optimal Markov policy and $V_N^*(i) = u_0^*(i)$ is the optimal value function. The algorithm terminates.

5: **else**

6: let $t := t - 1$ and turn to step 3;

7: **end if**

8: Step3: $\forall i_t \in S$, compute:

$$u_t^*(i_t) = \max_{a \in A(i_t)} \left\{ r_t(i_t, a) + \sum_{j \in S} p_t(j|i_t, a) u_{t+1}^*(j) \right\} \tag{2}$$

9: and record the set:

$$A_t^*(i_t) = \mathit{arg} \max_{a \in A(i_t)} \left\{ r_t(i_t, a) + \sum_{j \in S} p_t(j|i_t, a) u_{t+1}^*(j) \right\} \tag{3}$$

10: and select any of $f_t^*(i_t) \in A_t^*(i_t)$, then the decision rule at time t is defined as f_t^* .

11: Step4: turn to step 2

The set $A_t^*(i_t)$ is usually called optimal action set. The backward recursive iteration algorithm reflects the concept of **Principle of Optimality**, which is first presented in [5]. *Bellman's Principle of Optimality* is stated as: "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision." (see [4])

4.2 Forward Value Iteration

The backward value iteration is based on final reward. During the computation, the transition probability matrix and reward function is assumed not to change. But in dynamic and stochastic web service environment, web services selected by the policy may fail. If the iteration is modified to start from the beginning, web service invocation failure can be accommodated in the policy computation process as mentioned in Subsection 3.3.

The value iteration computing process is the same as the one introduced last subsection, with except that iteration equation is modified as:

$$V_0^*(i_N) = r_N(i_N) \quad (4)$$

$$V_t^*(i_t) = \max_{a \in A(i_t)} \left\{ r_t(i_t, a) + \sum_{j \in S} p_t(j|i_t, a) V_{t-1}^*(j) \right\} \quad (5)$$

5 Experiments

In order to evaluate the methods, experiments are performed to examine the performance of the methods discussed in this paper. The PC configuration: Pentium 4 1.5GHZ with 512M RAM, Windows 2000, jdk1.4.2.

To compare different methods, computation time is measured in experiments. In this process, the number of task nodes is varied from 10 to 100 by adding tasks randomly and the number of candidate services per service class is varied from 10 to 50, both with steps of 10. All experiments are conducted 10 times and the average computation cost is computed.

Abstract specifications of composite web services are defined using abstract services that can be implemented by a collection of concrete services by registering concrete web services with each abstract one. The detailed description of abstract and concrete web services are stored in Xindice(<http://xml.apache.org/xindice/>), while some of indexing columns such as service id, service name and quality parameters are stored in MySQL. This facilitates the process of identifying all candidate web services for a given abstract service. The QoS data are generated according to random variables following Gaussian distribution.

Fig. 2 presents the computation cost (in seconds) of MDP method using backward value iteration and forward value iteration (without failure). The computation cost is acceptable as far as the experiments are concerned. For example,

when there are 100 tasks and average 50 candidate services per class, the computation cost is about 1.07 seconds and 1.14 seconds for backward and forward iteration, respectively. Only in this case is the computation cost more than one second. This means the methods may be useful in dynamic service composition for such problem size.

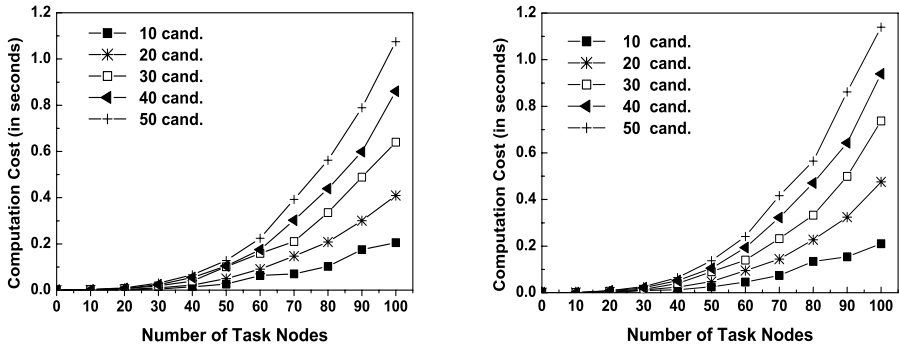


Fig. 2. Experimental results for value iteration algorithms. The left side is backward and the right side for value iteration(without failure).

The impact of web service invocation failure is also examined in experiments. Fig. 3 shows the computation cost (in seconds) of MDP method using value iteration with service invocation failure. The average failure probability of service invocation is about 5% and 10%, respectively. Though the cost shown in Fig. 3 is higher than that in Fig. 2, it is still acceptable. For example, when there are 100 tasks and average 50 candidates per class, the computation cost is about 1.42 seconds and 1.53 seconds with probability 5% and 10%, respectively.

Fig. 4 compares the computation cost for the three methods, where average failure probability is set to 5% for forward value iteration. It can be seen from the figure that the performance of backward method and value iteration without failure is almost the same. While the computation cost for value iteration with failure is higher than the other two methods. This can be explained that the system will stay at some state when failure happens and the number of iteration increases.

To further illustrate the impact of average failure probability of service invocation on computation cost, experiments are repeated for value iteration algorithm with different average failure probability. In experiments, the probability is set to 1%, 5% and 10%, respectively. Fig. 5 shows the results when the number of candidate services per service class is 20 and 50. From Fig. 5, it can be observed that computation cost increases when average failure probability increases. As the failure probability increases, the chances for the system to stay at some state increase as. So the average number of iteration increases. And according to the algorithms, optimal action is computed for every state during each iteration. So the increased number of iteration incurs more computation.

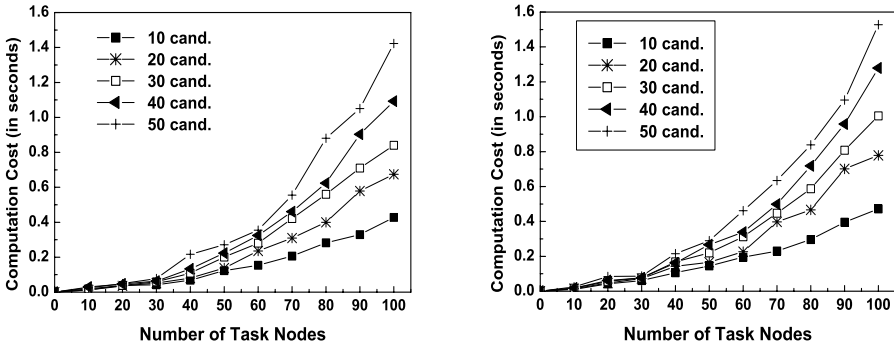


Fig. 3. Experimental results for value iteration algorithms(with web service invocation failure). The left side is for 5% and the right side for 10%.

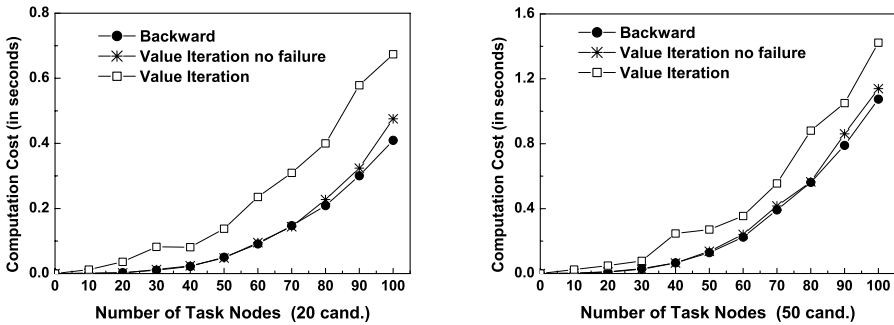


Fig. 4. Comparison of computation cost for three methods, where average failure probability is set to 5% for forward value iteration. The left side is for 20 candidates and the right side for 50 candidates.

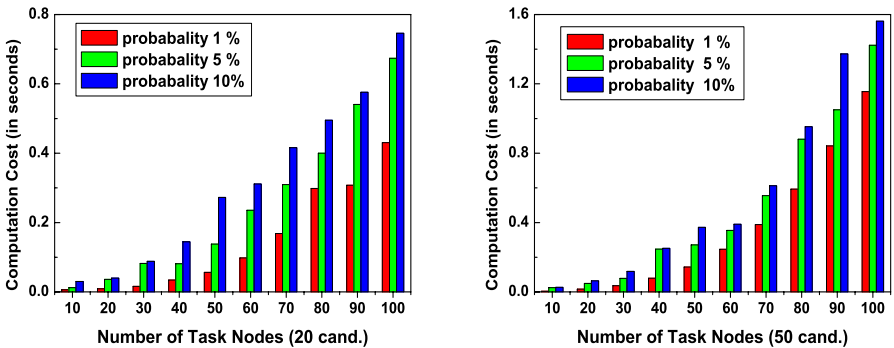


Fig. 5. Performance of Value Iteration with different Failure Probability. The left side is for 20 candidates and the right side for 50 candidates.

6 Related Works

The emerging paradigm of web services promise to bring to distributed computation and services the flexibility that the web has brought to the sharing of documents. Related works on formal models of web services , automatic composition, analysis and verification can be found in [1, 2, 15, 16].

In industry, BPEL4WS(Business Process Execution Language for Web Services) [6] is the most popular standard for web service composition. However, performance and QoS are not covered in BPEL4WS. The topic of web service QoS and performance is discussed in [17–21]. Dynamic web service selection and composition is covered in [12, 11, 22]. Authors in [17] discussed a method for capacity planning for web services using Queueing Network-based models. QoS issues about web services are in [18, 19] covers quality computation and QoS negotiation. QoS requirements for web services are defined in detail in [20, 21]. Quality definition of web service in this paper is the same as previous works, which can be found in [11, 12].

This paper is most related to [3] which presents a policy-based approach for dynamically choreographing web services. The differences between this paper and [3] are the followings. (1)In contrast to [3], this work is a general framework for representing web service composition using MDP. While the stochastic nature shown by the motivating example in [3] is inherently a domain specific business logic. (2) Web service composition constructs are discussed in this paper, which makes this method flexible and can be easily extended to accommodate other composition constructs. Some frequently used composition constructs are discussed, including sequential, parallel, conditional and iterative. (3) In addition to forward value iteration method, backward value iteration algorithm for MDP is also investigated in this paper. (4) The performance of value iteration algorithms is examined by experiments while the performance issue is not mentioned in [3]. (5) This paper explores the impact of service invocation failure probability on the performance of value iteration algorithm. (6) MDP model in this paper is based on QoS criteria, which is the promising base for web service publishing, brokering, billing, contracting and composing according to [2].

7 Conclusion

Markov decision process is used to model the process of selecting web services dynamically in this paper, where N horizons total expected reward is the criterion. It is defined on the base of QoS description for web services. Web service composition patterns including sequential, conditional, parallel and iterative are modeled in MDP. Two algorithms are introduced and implemented for computing optimal policy. Experimental results show that these methods are efficient and it is acceptable of the computation cost. In future work, other criteria for Markov decision process will be employed such as infinite horizon expected total discounted reward and average reward, etc. Composition constructs presented in BPEL4WS ([6]) will also be further investigated in future work.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No. 90412010 and ChinaGrid project of the Ministry of Education, China.

References

1. R. Hull, M. Benedikt, V. Christophides, and J. Su. E-services: A look behind the curtain. In Proc. ACM Symp. on Principles of Database Systems, 2003.
2. Aphrodite Tsalgatidou, Thomi Pilioura, An Overview of Standards and Related Technology in Web Services, Distributed and Parallel Databases, 12, 125-162, 2002
3. Prashant Doshi, Richard Goodwin and Rama Akkiraju, Kunal Verma, Dynamic Workflow Composition using Markov Decision Processes, In proceedings of the IEEE International Conference on Web Services (ICWS2004)
4. Liu Ke Applied Markov Decision Processes, Beijing: Tsinghua Press 2004, page 21 (in Chinese)
5. Bellman R. E. Dynamic Programming. Princeton University Press, Princeton, New Jersey, 1957
6. Business Process Execution Language for Web Services, version 1.1, <http://www.ibm.com/developerworks/library/ws-bpel/>
7. WS Choreography Model Overview, <http://www.w3.org/TR/ws-chor-model/>, 2004
8. OWL Services Coalition. OWL-S: Semantic markup for web services, November 2003.
9. W3C, "Web Services Description Language (WSDL) Version 2.0", W3C Working Draft, March 2003. (See <http://www.w3.org/TR/wsdl20/>.)
10. Wil M.P. van der Aalst, Marlon Dumas, Arthur H.M. ter Hofstede Web Service Composition Languages: Old Wine in New Bottles? In: Gerhard Chroust and Christian Hofer eds. Proceedings of the 29th EUROMICRO Conference "New Waves in System Architecture" (EUROMICRO'03), IEEE Computer Society, 2003, 298-305
11. Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, and Maria Luisa Vilani, A Lightweight Approach for QoS-Aware Service Composition, 2nd International Conference on Service Oriented Computing (ICSOC), New York: ACM Press, 2004, 63-63
12. Liangzhao Zeng, Boualem Benatallah, Anne H.H. Ngu, Marlon Dumas, Jayant Kalagnanam, Henry Chang, QoS-Aware Middleware for Web Services Composition, IEEE transactions on Software Engineering, 2004, 30(5):311-327
13. D.J. White Markov Decision Processes, Wiley 1993, Page 1-96
14. Qiyang Hu, Jianyong Liu, An introduction to Markov Decision Processes, Xidian University Press 2000 (in Chinese)
15. Richard Hull, Jianwen Su Tools for Design of Composite Web Services (ppt), <http://www.cs.ucsb.edu/su/tutorials/sigmod2004.htm>
16. A. Deutsch, L. Sui, and V. Vianu. Specification and verification of data-driven web services. In Proc. ACM Symp. on Principles of Database Systems, 2004.
17. Dunlu Peng, Yang Yuan, Kun Yue, Xiaoling Wang, and Aoying Zhou, Capacity Planning for Composite Web Services Using Queueing Network-Based Models, WAIM2004, LNCS3129, pp.439-448, 2004

18. D.A. Menasc, "Composing Web Services: A QoS View," IEEE Internet Computing, 2004,8(6): 88-90
19. D.A. Menasc, H. Ruan, and H. Gomaa, "A Framework for QoS-Aware Software Components," Proc. 2004 ACM Workshop on Software and Performance, ACM Press,2004, pp. 186-196
20. Shuping Ran, "A Model for Web Services Discovery With QoS", ACM SIGecom Exchanges, March 2003, 4(1)
21. QoS for Web Services: Requirements and Possible Approaches, Hyderabad, 4 - 6 August 2004
22. Ghandeharizadeh, S. (et al.) "Proteus: A System for dynamically composing and Intelligently Executing Web Services". ICWS2003.

FECT: A Modelling Framework for Automatically Composing Web Services*

Lishan Hou^{1,2} and Zhi Jin^{2,3}

¹ Graduate School of Chinese Academy of Sciences

² Academy of Mathematics and Systems Science, CAS

{hlsalisa, zhijin}@amss.ac.cn

³ Institute of Computing Technology, CAS
Beijing 100080, P.R. China

Abstract. In this paper, we propose FECT, a new modelling framework for describing and composing heterogenous Web services to satisfy emergent requirements. In FECT, a three-dimension description pattern, i.e. the **Environment**, the **Capability** and the behavior **Traces**, has been proposed to describe Web services. FECT has the following features, which may distinguish it from other work. Firstly, the environment, with which Web services may interact, has been introduced for characterizing the context of Web services. Secondly, the capability of a Web service is captured as the effects on its environment. Thirdly, the composition process is accomplished by combining the traces of Web services.

1 Introduction

Web service is a kind of self-contained, Internet-enabled applications capable not only of performing activities on its own, but also possessing the ability to engage other Web services in order to complete higher-order activities. Seamless composition of Web services has enormous potential in streamlining business-to-business application integration [1]. But there are many inherent difficulties in this new paradigm of application development. Services distributed on Internet are usually developed by different organizations and described by different terms and in different ways. That makes the understanding and communication among them be difficult. As a result, the ability to dynamically and automatically discover, understand and compose heterogeneous Web services becomes a tough problem.

These problems, in fact, have been paid many attentions at present. The first effort is about the standards of Web service description, which are trying to provide sufficient information to enable automated discovery, composition, as well as execution. WSDL [2] is an XML format for describing services as a set of

* This work was partly supported by the National Natural Science Foundation of China (No.60233010 and No.60496324), the National Key Research and Development Program (Grant No. 2002CB312004) of China, the Knowledge Innovation Program of the Chinese Academy of Sciences and MADIS of the Chinese Academy of Sciences.

endpoints or ports operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly and bound to a concrete network protocol(SOAP [3], HTTP,etc.). WSDL rests on the communication level description of the messages and protocols used by a Web service. It defines and formats query interactions with a service, but doesn't provide a model for the semantics of such exchanges. OWL-S [4] takes the initiative to provide an ontology markup language expressive enough to semantically represent capabilities and properties of Web services. It provides three essential types of knowledge, i.e. *ServiceProfile*, *ServiceModel*, and *ServiceGrounding*, for a Web service. OWL-S focuses on the representation of what the service does rather than where to find the service.

Another effort worthy to be mentioned is on the composition of Web services. BPEL4WS [5] models a Web service as a business process from the industrial view. It defines the interaction protocols between a process and its partners; and also defines an inter-operable integration model that should facilitate the expansion of automated processes. As for formalization, Petri-net [6] and process algebra (e.g. CCS [7]) also contribute some tentative work to the composition of services.

Although there have been many efforts on these topics, the discovery and the composition of Web services still face big challenges. All of them model the capabilities of Web services only by input/output transformations. However, the semantic meaning of an application software is in fact the effect it can cause on the environment which the software interacts with [8][9][10]. In other words, the requirements for the software(i.e. a Web service here) are located in the reality, and in the environment. The relationship among environment(\mathcal{E}),requirements(\mathcal{R}) and software(\mathcal{S}) has been revealed as an entailment:

$$\mathcal{E}, \mathcal{S} \vdash \mathcal{R}$$

That means that if a Web service(\mathcal{S}) is installed in the environment described by \mathcal{E} , then the environment will exhibit the properties described in \mathcal{R} . Although the relationship is not an implication,it captures the essence that \mathcal{R} can be satisfied if a Web service(\mathcal{S}) is installed in the environment(\mathcal{E}). From this sense, the environment and the effect on it caused by the Web service should be two core aspects to catch the meanings of the Web service.

We propose a new modelling framework for describing, discovering and composing Web services automatically. This approach is based on the rationales that the semantics of Web services can be grounded in the environment which Web services are situated in and interact with. The rest of this paper is organized as follows. Section 2 introduces the architecture of our FECT (Framework based on Environment, Capabilities and behavior Traces) approach. The description of Web services based on *environment ontology* and *interaction ontology* in FECT is defined in section 3. Section 4 designs the mechanism for the selection and composition of Web services and illustrates the whole process by a trial case. Section 5 reviews some related work and concludes this paper.

2 FECT Architecture

The overall architecture of the FECT is shown in Figure 1. The main components are the Ontology Base, the Service Registry and the FECT Server.

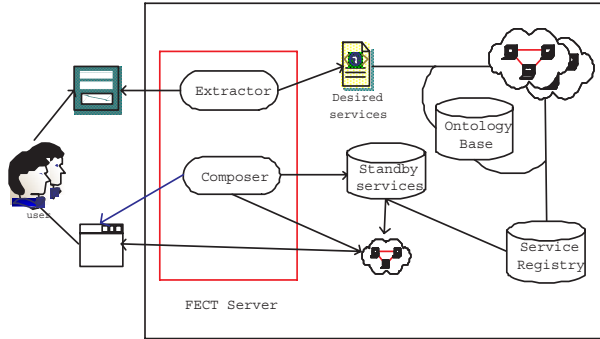


Fig. 1. Architecture of composition framework

The Ontology Base provides the description of concept terms, associations and constraints for the environments and interactions used in particular problem domains. It is a sharable conceptualization to support the recognition and understanding of existing published services as well as the emergent request of new Web services.

The Registry is a special service whose responsibility is to organize the Web services. A service representation in the Registry contains a pointer to the provider of the service and a port where to access the service like in UDDI[11]. Furthermore, it also includes the declaration on the effect, i.e. the changes of the environment caused by the service, and interaction behavior traces with the environments. Web services can be published and recognized only if they follow the standards of the Registry. The detailed description of services will be discussed in section 3.

The FECT server contains two parts: the requirements extractor and the service composer. The requirements extractor elicits the external requirements, models them by using the automated modelling techniques[12][13], and publishes the models on Internet as desired Web services based on the domain ontologies. Once the desired service-models occur, those autonomous Web services which have the ability to complete the tasks or sub-tasks of the requirements models, volunteer to be the candidates. All of these candidates construct a *standby service* base. Then the service composer combines some services in the base to form short-term partnerships in order to satisfy the emergent requirements.

3 Description of Web Services and Requirements

Appropriate description of the capability is the prerequisite of understanding requirements as well as Web services. Most of the current work in this field

only depend on the description in syntax level. We argue that adaptive and intelligent discovery and composition of Web services can be achieved only after the meaning of the requirements has been well understood and the semantic model of Web services has been built.

In order to bridge the gap between requirements and Web services, we use the “environment” the Web services exist in, “effects” of the environment caused by the Web services and the “interactions” of the Web services with the environment, to capture the meanings of the Web services and the requirements. For a Web service, its environment contains the controllable resources which the service visits and acts on, and the incontrollable outer triggers from people, other autonomous Web services, time, etc. The published capabilities of the Web services and the desired capabilities of the requirements are designated on two ontologies, i.e. the domain environment ontology and the environment interaction ontology. These ontologies bring the semantic meanings into the capability description.

3.1 Environment Ontology and Interaction Ontology

The incontrollable triggers are uncertain because of the reactivity of the Web services and the instability of the Internet platform. The environment ontology can be represented in a hierarchy of controllable resources together with the effects on them. Figure 2 shows the top level of the resources in the environment ontology of the Web sale domain.

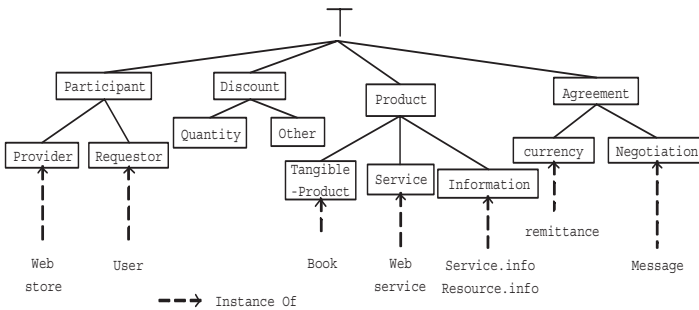


Fig. 2. A hierarchy of resources of the web sale domain

In the environment ontology, each individual resource has one or more attributes to describe itself. These attributes are divided into two groups: the static attributes, i.e. their values are fixed once the attributes are instantiated, and the dynamic attributes, i.e. their values are changeable along different situations. We call the static attributes the “information” of the resource (“info” for short) and call the dynamic attributes the “states” of the resource. We use the state transition graph to stipulate the permitted state changes. All the possible state changes by a Web service just reflect its capabilities. For example, Figure 3

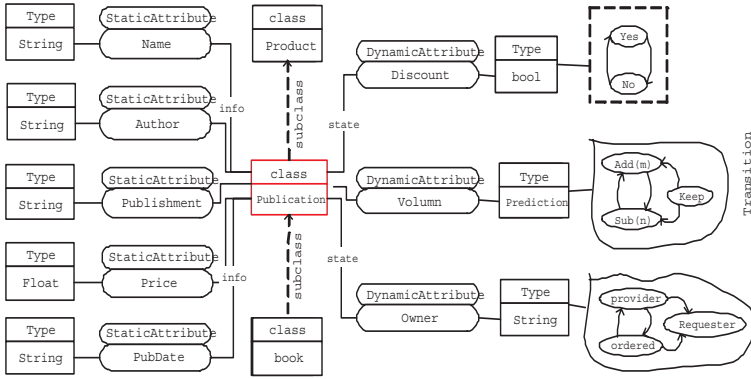


Fig. 3. Ontology description to "book" of the web sale domain

is a part of resource descriptions, in which, *book* is a kind of resources in the Internet-based book-selling domain.

In the software world, the product-selling process is in fact transmitting the information and managing the product's state transitions. In this sense, the interaction ontology is designed to specify the interaction between the services and their environments. We assume that an interaction be an atomic behavior. Therefore a sequence of interactions can implement a macro-level or worthy to be observed behavior, denoted as "function" in the paper. Table 1 shows a fraction of the associations among the concepts in the interaction ontology.

Table 1. A fraction of the interaction ontology

SN	Source concept	Association	Target concept	Description
2	read	MethodOf	collect	"read" is one of the methods to fulfill the task "collect"
3	(ask,get)	MethodOf	collect	"(ask,get)" is one of the methods to fulfill the task "collect"
4	subscribe	MethodOf	collect	"subscribe" is one of the methods to fulfill the task "collect"
18	subscribe	$\equiv_{collect}$	receive	$\equiv_{collect}$ is an equivalence relation, "subscribe" and "receive" are replaceable when fulfilling the task "collect"
22	write	precondition	read	"write" is necessary before "read" is invoked
24	inform	precondition	receive	"inform" is necessary before "receive" is invoked

3.2 Behavior Trace

Besides environments, another feature of FECT is viewing each service (or each piece of requirements) as a process with some kind of interaction behaviors.

CSP [14] expresses a process as a finite sequence of symbols recording the events in which the process has engaged up to some moment in time. The same idea is also suitable to the description of the interaction sequences of Web services.

Definition 1. (Behavior Trace) A sequence $\langle x_1, x_2, \dots, x_n \rangle$ is called a behavior trace of a service if x_1, x_2, \dots, x_n form a behavior sequence in the interaction process.

Here, each $x_i (1 \leq i \leq n)$ is not an event as that in CSP, but a trigger interaction ω^1 or a single interaction transferring information from the service to its environment or from its environment to the services. Formally,

$$x_i = \omega \text{ or } P(r), \quad 1 \leq i \leq n$$

where P is a behavior concept in the interaction ontology and r is the resource operated by P .

Example 1. The trace for ATM verifying the validity of credit cards can be denoted as: $\langle \text{ask}(\text{password}), \omega, \text{get}(\text{password}) \rangle$.

It is necessary to define some operations on the behavior traces and some relations among them. These operations are a bit different from those in CSP.

Definition 2. (Catenation \wedge) Let $X = \langle x_1, x_2, \dots, x_n \rangle$ and $Y = \langle y_1, y_2, \dots, y_m \rangle$ be two behavior traces, then

$$X \wedge Y = \langle x_1, \dots, x_n, y_1, \dots, y_m \rangle$$

is called catenation of X and Y , which means that X takes place before Y .

Definition 3. (Trace Ordering \leq) Let X and Y be two behavior traces, $X \leq Y$ if $\exists Z = \langle z_1, z_2, \dots, z_l \rangle$, such that $X \wedge Z = Y$.

Obviously, \leq is a relation of partial ordering, i.e. it is reflexive, antisymmetric and transitive.

Definition 4. (Behavior Ordering \preceq) Let behaviors $X = \langle x_1, x_2, \dots, x_n \rangle$ and $Y = \langle y_1, y_2, \dots, y_m \rangle$ be two traces, for x_i in X and y_j in Y , $y_j \preceq x_i$ if y_j takes place before x_i .

Definition 5. (General Catenation Δ) Let $X = \langle x_1, \dots, x_n \rangle$ and $Y = \langle y_1, \dots, y_m \rangle$ be two behavior traces, there exists a group of behavior orderings

$$\{y_{i_k} \preceq x_{j_k} | y_{i_k} \text{ is in } Y, x_{j_k} \text{ is in } X\}_{p < n, m}$$

in which the smallest index of x is denoted as j_{k0} and corresponding index of y is i_{k0} . Then the general catenation of X and Y is defined as follows:

$$X \Delta Y = \begin{cases} X \wedge Y & p = 0 \\ \langle x_1, \dots, x_{j_{k0}-1} \rangle \wedge \langle y_1, \dots, y_{i_{k0}-1} \rangle \wedge \langle y_{i_{k0}}, x_{j_{k0}} \rangle \\ \wedge \langle x_{j_{k0}+1}, \dots, x_n \rangle \Delta \langle y_{i_{k0}+1}, \dots, y_m \rangle & p \geq 1 \end{cases}$$

¹ ω is used to denote the outer trigger from people, other autonomous Web services, time, etc. in the interaction.

” Δ ” is a kind of catenation which keeps the underlying behavior orderings. It can be used widely in composing Web services.

3.3 Description Framework

It is time to give descriptions of the services as well as the requests. In order to facility the matchmaking, we adopt the following coincident description elements.

- **Resource**, the controllable environment which the service interact with. It is modelled as two groups of attributes.

$$Res = \{r|r \models r.info \cup r.state\}$$

Here r is a sort of resources, $r.info$ and $r.state$ are the sets of r 's static and dynamic attributes;

- **Effect**, the transitions of the resource states caused by the service.

$$Effect = \{eff \models r.attr(v_1, v_2, P \text{ or } X)|r \in Res \wedge attr \in r.state\}$$

That means that the value of $r.state$ changes from v_1 to v_2 by reason of the interaction(s), P is a single behavior in the interaction ontology and X is a behavior sequence. Effect reflects the capability of the service;

It is worthy to note that the transitions in $Effect$ is transitive, i.e. if

$$r.attr(v_1, v_2, P_1) \in Effect \text{ and } r.attr(v_2, v_3, P_2) \in Effect$$

then

$$r.attr(v_1, v_3, < P_1 > \wedge < P_2 >) \in Effect$$

- **Action**, the interaction sequences of the service with its environment.

$$Action = \{X_1, \dots, X_n\}$$

In which $X_i(1 \leq i \leq n)$ is a behavior trace.

- **Function**, worthy to be observed capability of the service, which is implemented by a sequence of interactions.

$$Func = \{FP \models X \text{ or } P|X \text{ is a behavior trace, } P \text{ is an interaction}\}$$

The definition of FP rests with the particular domain knowledge.

These elements characterize the service from multi-facets: environment, effects on the environment and interactions with the environment. The published services show their information by the following four-slot frame.

Frame: name of the published Web service	
Res	controllable environment of the service
Effect	transitions of the resource states caused by the Web service
Action	interactions between the service and its environment
Func	worthy to be observed functions of the service

Due to different focuses, requirements are described by *Res*, *Effect*, and *Action*. The requirement is modelled as a group of sub-services in terms of goals, tasks, etc. and published onto Internet. We term these sub-services the *desired services* because they might be the best solution we expect for the current requirements. The desired services are described with a three-slot frame(*Res*, *Effect*, *Func*).

The XML-based abstract description can be represented as:

```

<service>
  xmlns: Res="http://www.fect.com/environment/resource/"
  xmlns: Effect="http://www.fect.com/environment/resource/"
  xmlns: Action="http://www.fect.com/ontology/interaction/"
  xmlns: Func="http://www.fect.com/ontology/interaction/"
  <Res> < r1 > <info (attrm, ...)/ > <state (attrn, ...)/ > < /r1 > ... </Res>
  <Effect> (r.attr(v1, v2, P), ...) </Effect>
  <Action> <trace> << x11, ... x1n >>2 </trace> ..... </Action>
  <Func> (FP1, FP2, ...) </Func>
</service>

```

It is tellable that the above XML-format leaves out the description of the WSDL interface and SOAP protocol in order to emphasize the semantic representation of the Web service.

4 Composition of Web Services

This section uses a meaningful example, i.e. book-selling on Internet, to illustrate the four-step procedure of discovering and composing Web services in FECT.

Step 1: Extract Requirements from the Real World

Suppose that a request service *T*, to build and manage a Web-based bookstore, has been put forward in the following form.

```

<Requirement> xmlns: .....
  <Res> (book)</Res>
  <Effect> book.owner(store, user, ω) </Effect>
  <Action>
    <trace> << register, login, order, pay >> </trace>
    <trace << login, logout >> /> <trace << order, cancel >> />
  </Action>
</Requirement>

```

² For avoiding the symbol conflicts of trace and XML, we substitute “<< >>” for “< >” when describing services by XML.

Step 2: Model the Requirements and Publish Them on Internet

After getting the request, the FECT server models it as three desired sub-services by the role-based modelling techniques[13]: *user-manager*, *book-manager* and *order-manager*. It then publishes them as the desired services in XML style.

```

<DesiredService1>
<Res> <user>
    <info>(Name, Address, Contact, Password)</info>
    <state>(Authority)</state>
</user> </Res>
<Effect> (user.Authority(anonymous, legal, collect(user.info)), user.Authority(legal,
    online, collect(user.password)), user.Authority(online, anonymous, ω)) </Effect>
<Func>(collect(user.info), provide(user.info),alter(user.Authority))</Func>
</DesiredService1>
<DesiredService2>...</DesiredService2>
<DesiredService3>...</DesiredService3>

```

Step 3: Select the Competent Ones from the Volunteer Services

As autonomous softwares, once recognizing the newly published desired services, the Web services decide whether they can match the descriptions of the desired services and whether they are willing to join in as members according to the rules of themselves. At the same time, the FECT server has also a group of rules to evaluate whether the desired services and the volunteer services are suited. It selects those competent ones as candidates to form the Standby Services Base.

Let s_1 be a Web service, s_2 be an desired service. How to judge whether s_1 is competent for s_2 ? Here we give the following three qualifications:

1. If there exist $r \in s_2.Res$ and $r' \in s_1.Res$ such that the following statements hold.
 - (a) $r = r'$ or $SubClass(r, r') = True$. Means r is just r' or a subclass of r' .
 - (b) $r.info \subseteq r'.info$. Means the information of r is subset of that of r' .
 - (c) $r.state \subseteq r'.state$. Means the set of states of r' contains that of r .
2. s_1 and s_2 have some effects in common, i.e. $s_1.Effect \cap s_2.Effect \neq \emptyset$
3. For any interaction $fp_1 \in s_1.Func$, there exists $fp_2 \in s_2.Func$, such that at least one of the following conditions holds:
 - (a) $fp_1 = fp_2$
 - (b) $MethodOf(fp_1, fp_2) = True$
 - (c) $fp_1 \equiv_T fp_2$ (That means fp_1 and fp_2 are replaceable mutually when dealing with the task "T".)
4. For any trace $FP_1 \in s_1.Func$, $\exists FP_2 \in s_2.Func$, such that $FP_1 = FP_2$.

According to the above judgement standards, the services who satisfy 1, 2 and at least one of 3,4 are regarded as candidates. FECT found the following suitable services for the book-selling example.

```

<UserManager001>
<Res> <user>
    <info (Name, Address, Contact, ID, Age) /> <state (Authority, VIP) />
</user> </Res>
<Effect>
    (user.Authoriy(free,legal, <ask,get>(user.info)), user.Authority(legal, online,
    <ask,get>(user.info)), user.Authority(online, legal, ω),user.Authority(legal, free, ω))
</Effect>
<Action>
    <trace> «ask, ω, get, add, EDask, tell»(user.info)</trace>
    <trace> «ω, alter(user.Authority) » </trace>
</Action>
<Func>
    (<ask, ω, get >(user.info),<Edask,tell>(user.info), alter(user.Authority),
    add(user.info),delete(user.info))
</Func>
</UserManager001>
<BookManager001>...</BookManager001>
<OrderManager001>...</OrderManager001>
    
```

Step 4: Compose the Chosen Services

Now, suppose that Standby Services Base contains $\{WS_1, WS_2, \dots, WS_m\}$. Whether they can cooperate with each other to satisfy the requirement? The composition of services can be regarded as a rotative process:

- (1) select two composable services WS_i and WS_j and compose them as a new Web service WS ;
- (2) update the Standby Services Base by replacing WS_i and WS_j with WS .

The eventual composite services in the Standby Base are the concurrent processes to implement the requirements together.

Before presenting the algorithm for composing two Web services, we introduce \oplus to denote the composition operators of resources, effects, actions and even Web services.

- $Res_1 \oplus Res_2 \stackrel{\text{def}}{=} Res_1 \cup Res_2$. This is the simplest case.
- $Effect_1 \oplus Effect_2 \stackrel{\text{def}}{=} Effect_1 \cup Effect_2 \cup Effect_{1+2}$, in which each element of $Effect_{1+2}$ is a new member because of the transitivity of the elements in $Effect_1 \cup Effect_2$.
- $Action_1 \oplus Action_2 \stackrel{\text{def}}{=} Action_1 \cup Action_2 \cup Action_{1+2}$, in which each element of $Action_{1+2}$ is a catenation of the two traces from $Action_1$ and $Action_2$ respectively.

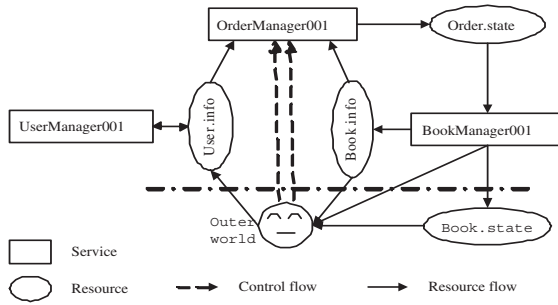


Fig. 4. Composition result of Web bookstore

5 Related Work and Conclusions

Apart from DAML-S and BPEL4WS, there are many other schools in the field of composing Web services. WSCI [15] provides mechanisms for describing how Web services can be assembled to participate in the shared business processes. EFlow [16] models a composite service by the graph that defines the order of execution among the nodes in the process. The graph is created manually and updated dynamically. Self-Serv [17] develops a relatively complete but a bit fuzzy middle-ware infrastructure for the composition of Web services. IRS-II [18], a framework and implemented infrastructure, bridges requirements and Web services by mapping the task models and the PSMs(Problems Solving Methods). However, all of these thoughts are absorbed in the composition at the execution level.

FECT is a framework to model the automatic composition of Web services based on domain ontology and behavior trace. Compared with the related work, FECT has its own significant features. It is the first try to make the services self-described by using the effect on their environments and the interaction traces. In nature, FECT introduces a bottom-up and automated composition process of Web services via the combination of behavior traces based on domain ontologies. At this stage, many further works are needed for improving FECT, including establishing meaningful meta-ontology, formalizing the service description, and developing the verification method of composite services.

References

1. J. Koehler and B. Srivastava. Web service composition: Current solutions and open problems. In *ICAPS 2003 Workshop on Planning for Web Services*, pages 28–35, 2003.
2. Web services description language (WSDL) 1.1. <http://www.w3.org/TR/wsdl>, 2001.
3. SOAP version 1.2. <http://www.w3.org/TR/soap/>, 2004.
4. A.Ankolekar et al. DAML-S: Semantic markup for web services. *Proceedings of the International Semantic Web Workshop*, 2001.

5. S.Weerawarana. Business process with BPEL4WS: Understanding BPEL4WS. <http://www-106.ibm.com/>, 2002.
6. R. Hamadi and B. Benatallah. A petri net-based model for web service composition. In *Proceedings of the 14th Australasian Database Conference (ADC'03)*, pages 191–200. Australian Computer Society, February 2003.
7. Robin Milner. *A Calculus of Communicating Systems-Lecture Notes in Computer Science*. Number 92. Springer-Verlag, 1980.
8. D.L.Parnas and J.Madey. Functional documentation for computer systems. *Science of Computer Programming*, 25(1):41–61, 1995.
9. M.Jackson. The meaning of requirements. *Annals of Software Engineering*, 3:5–21, 1997.
10. M.Jackson. *Problem Frames: Analyzing and structuring software development problems*. Addison-Welsley, 2001.
11. UDDI executive overview: Enabling service-oriented architecture. <http://uddi.org/pubs/>, 2004.
12. Z.Jin, D.A.Bell, F.G.Wilkie, and D.G.Leahy. Automated requirements elicitation: Combining a model-driven approach with concept reuse. *International Journal of Software Engineering and Knowledge Engineering*, 13(1):53–82, 2003.
13. Eric SK.Yu. Towards modelling and reasoning support for early-phase requirements engineering. pages 6–8, Washington D.C., USA, Jan. 1997.
14. C.A.R.Hoare. *Communicating Sequential Processes*. Prentice Hall Internationl, 1985-2004.
15. Web service choreography interface (WSCI) 1.0. <http://www.w3.org/TR/wsci>, 2002.
16. F.Casati et al. Adaptive and dynamic service composition in eflow. In *Proc. of the International Conference on Advanced Information Systems Engineering*, 2000.
17. B.Benatallah, Q.Z.Sheng, and Marlon Dumas. The self-serv environment for web services composition. *IEEE Internet Computing*, 7(1):40–48, 2003.
18. Enrico Motta¹, John Domingue¹, and Liliana Cabral¹. Irs-ii: A framework and infrastructure for semantic web services. In *Proceedings of the 2nd International Semantic Web Conference 2003 (LNCS)*, volume 2870, pages 306–318, Florida, USA, October 2003. Springer-Verlag GmbH.

Intrusion Detection of DoS/DDoS and Probing Attacks for Web Services

Jun Zheng and Ming-zeng Hu

The Research Center of Computer Network and Information Security Technique,
P.O.Box 320, Harbin Institute of Technology, Harbin, China, 150001
{zhengjun, mzhu}@hit.edu.cn

Abstract. The (Distributed) Denial of Service (DoS/DDoS) attacks have become the main devastating threats to web services, and generally, the Probing attacks are the prior steps of DoS/DDoS attacks. To achieve the aim of the information assurance, an intrusion detection mechanism based on the Vector Quantization (VQ) technique is proposed for countering DoS/DDoS and Probing attacks in this paper. The normal network traffic usage profile can be modeled and represented by the codebook of VQ from which the abnormal behavior deviation of TCP traffic can be measured quantitatively well. In data processing, according to the characters of DoS/DDoS and Probing attacks, we implement the novel feature extraction of TCP flow state. We apply the detection mechanism to DARPA Intrusion Detection Evaluation Data Set. It is shown that the network attacks are detected with more efficiency and relatively low false alarms.

1 Introduction

With the ever-fast development of the Internet, all kinds of network service are rising today. For example, the Web service becomes more and more popular and general in the Internet. Because of the extensive public available, the Web service also becomes the main object of malicious attacks, especially when the e-business flourishes. To strengthen the security of Web servers and connected database, in addition to security techniques, such as access control policy, authentication and encryption, Intrusion Detection System (IDS) becomes an important and traditional security-barrier against network-based attacks.

Among the malicious attacks, maybe DDoS/DoS and Probing attacks are the main threat. The DDoS/DoS attacks destroy the service availability directly, whereas Probing attacks usually are used to explore the detail information of servers, for example, the network Worms usually utilize probing attacks to scan the certain scope network using some given scanning policy. Both of two attacks are the TCP attacks mainly. According to the statistical data from Moore [1], the majority of DoS/DDoS attack which is main threat to the whole Internet is deployed by using TCP as 90~94%. In this paper, we concentrate on the TCP attacks via extracting the TCP header information.

There are two general approaches in Intrusion Detection: Misuse Intrusion Detection (MID) and Anomaly Intrusion Detection (AID). Similar to virus detection, misuse detection is based on the pattern matching to hunt for the signatures extracted

from the known attacks. However, AID constructing the normal usage behavior profile, named historical or long-term behavior profile. And the analysis model looks for deviations of the short-term behavior profile from the normal. The deviations can be treated as the baselines of estimating the attack activities from normal behaviors. The Fig.1 describes the relation and contrast of main logical procedures in MID and AID. The two approaches are contrary as well as unitive some time.

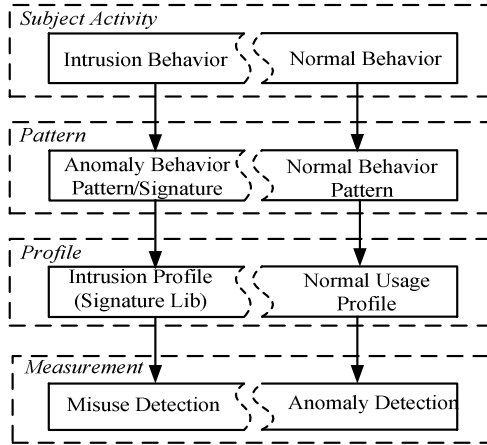


Fig. 1. Logic procedures in Intrusion Detection

In this paper, we propose a new AID mechanism using the Vector Quantization (VQ) [2]. The normal network traffic usage profile can be modeled in the codebook of VQ from which the abnormal behavior deviation can be measured quantitatively well. Furthermore, in data processing, according to the characters of DoS/DDoS and Probing attacks, we implement novel feature extraction of TCP flow state to achieve the maximal abnormal attack characteristics. The evaluation bed used in this paper is the 1999 DARPA Intrusion Detection Evaluation Data Sets.

The rest of paper is organized as follows: Section 2 explains the background and related work of AID, Section 3 proposes the new framework to construct the usage behavior profile and detect anomalies, and Section 4 illustrates the new method of data processing to intrusion detection. Section 5 describes the details of our experiments. Finally, Section 6 gives conclusions.

2 Background and Related Work

With the knowledge of normal behavior to detect novel intrusions, AID searches for anomalous behavior or deviations from the established baseline. If the deviations exceed the specified threshold of baseline, the activities will be flagged as anomalous. It is no doubt that the most important thing is to establish the profile characteristics of normal activity. Generally, the AID is looked as the binary classification: the given data sample is judged as normal activity or intrusion activity according to the AID model.

The most classical AID model comes from the earlier landmark paper of Dorothy Denning -An Intrusion Detection Model. [3] Denning describes building an "activity profile" of normal usage over an interval of time and proposed to employ statistics to construct a point of reference for normal behavior. Denning's work inspires the *statistical methodology* in Anomaly/Intrusion Detection fields.

However, as the attacks and intrusions become more complex, more and more *machine learning model and data mining model* are proposed to solve the problems of network security, including general Clustering [4], Outlier Detection, Support Vector Machine [5], Hidden Markov Models [6] and Neural Network [7] and so on. However, because of the complexities of algorithms, one of the main universal shortcomings of these methods is that these methods are not enough efficient to detect by the real time style. Without maintaining the profiles of normal behavior, Clustering and Outlier Detection [4] seem to be promising recently but these methods depend mightily on the precondition that abnormal data must take the little percentage among all the audit data for the certain temporal space. These methods can not get higher detection rate in case of DDoS or probing attacks because of the huge amount of activities and the higher proportion attack traffic during some certain time interval. Other approaches try to construct a certain profile according to different data objects in either host computer scope or network scope just as Table 1. SVM can achieve the higher classification rate but its algorithm complexity to describe the normal usage profile embarrasses the efficiency and usability in actual application. The same problems arise to HMM and Neural Network, especially in case of the huge amount of network traffic audit data in the network-based intrusion detection.

Table 1. Profile classification

Data Object	Profile	Scope
System Audit Log (e.g. BSM)	System Profile	Host Based
OS System Call	System Call Profile	
User Command Sequence	User Command Profile	
Network Traffic (Packet/Connection/Flow)	Network Traffic Profile	Network Based

In our methodology, we concentrate on AID and propose a novel approach to establish structural profiles of normal network traffic usage using VQ, and detect anomalies according to the deviations. Hereon, a codebook of VQ serves as the structural depiction of normal network traffic usage profile just deployed in section 3. However, our main aim is to implement a lightweight AID system based on the Quantization Vector to cooperate with other Misuse Intrusion Detection systems (e.g. SNORT [8]).

3 Vector Quantization for Intrusion Detection

Vector Quantization (VQ) [2] is an efficient coding technique, especially in data compressions of image and speech signal based on *the similarity measures between*

feature vectors. Through the compression approach of VQ, it is easy to transmit and to construct the index structure for the multimedia data files with high-dimensions. Traditionally, the initialization of Vector Quantization can be looked as the technology of the input space partition according to the similarities of input feature vectors. In this way, Vector Quantization is an approach to data clustering of a data set.

◆ **Definition1.** Vector Quantization can be defined as a mapping function from Euclidean space R^k into a certain finite subset C :

$$Q : R^k \rightarrow C, \quad C = \{y_0, y_1, \dots, y_{n-1} \mid y_i \in R^k\}. \tag{1}$$

The representative vector y_i in C is called **codewords**, and C is called **codebook**.

◆ **Definition2.** Given he input vector $x = (x_1, x_2, \dots, x_k)$ and the code-word $y_i = (y_{i1}, y_{i2}, \dots, y_{ik})$, every input vector will be assigned with a code-word in which the distortion between this codeword and the input vector is smallest among all codewords. The distortion is defined as **Quantization Errors** D :

$$D = \|x - y_i\| = [\sum_{j=1}^k (x_j - y_{ij})^2]^{1/2}. \tag{2}$$

The potential of VQ application to the usage profile analysis of network traffic is that VQ can compartmentalize the traffic feature vector space to groups comparing the similarities of feature vectors. All the profiles can be distilled and recapitulated into the codebook. Consequently, through the codebook organization of VQ, we can get the usage profiles to character the network traffic.

◆ **Definition3.** The usage profiles P of network traffic can be defined:

$$P_i = \{x \in R^k : Q(x) = y_i\}, i = 1, 2, \dots, n. \tag{3}$$

All x with the profiles P_i can be represented by y_i . Here, we can understand that VQ is also can be looked as a clustering processing that input space is grouped into N clusters.

3.1 Codebook Design

The first important step to implement Vector Quantization, codebook construction is the training processing by the normal network traffic with a certain algorithm.[2] The LBG algorithm [9] and Competitive Learning [10] are the most widely used approach to design codebook of VQ. Here, we use the Competitive Learning --Kohonen Learning Algorithm (KLA) [11] to train the structural codebook of normal network traffic. The codebook can be treated as the dictionary of normal network traffic behavior where the similar traffic behaviors are clustered to the same Voronoi partition space [12] represented by one codeword, also known as the centroid. After training, the normal traffic behaviors have been compacted to the codebook and the similar behaviors have been clustered together. The usage profiles are embodied in the codebook at last.

Table 2. Parameters in KLA

Parameter	Value
Topology	Hexa
Neighborhood function	Bubble
Dimension (X×Y)	30×30 40×40
Learning rate function	Linear
Training rate of first phase	0.5
Radius in first phase	20
Training length of first phase	50000
Training length of second phase	500000
Training rate of second phase	0.02
Radius in second phase	10

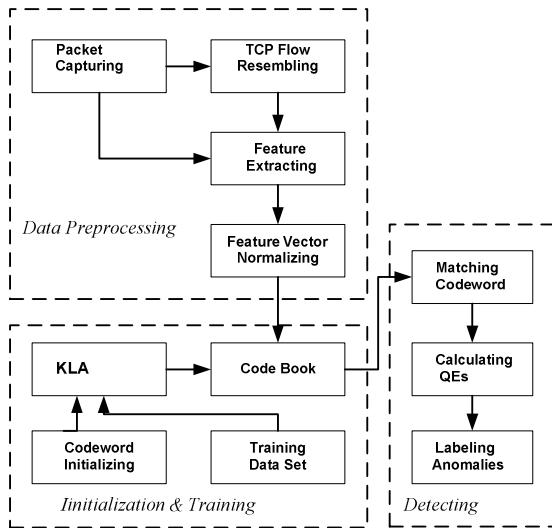


Fig. 2. Vector Quantization framework for intrusion detection

The codewords close to the input vectors by iterations. The codebook construction processing is the coupling processing between the all codewords and the input vector of network traffic.

◆ **Definition4:** Every TCP Flow is a data point in the n -dimension feature space R^n and R^n is Euclidian space.

$$TCPFlow = \{X | X \in R^n\}. \tag{4}$$

Every TCP Flow is expressed by the form of feature vector:
 $X = (x_1, x_2, \dots, x_n)$.

The following are the main steps involved in KLA:

The input vector: $X = (x_1, x_2, \dots, x_n)$ The codeword: $W = (w_1, w_2, \dots, w_n)$

Step1. To initialize every codeword of KLA with random values: $W_j(0)$

Step2. To compute the distance between the input vector X_i and the codeword $W_j(t)$, designate the winner neuron node j^* with the smallest distance. j^* is also called the **Best Matching Unit (BMU)**.

$$j^* = \arg \min_{1 \leq j \leq m} \|X_i - W_j(t)\| \tag{5}$$

The Euclidean distance is chosen as **Quantization Errors (QEs)**:

$$D = \|X_i - W_j(t)\| = \left(\sum_{k=1}^n (x_{ik} - w_{jk}(t))^2 \right)^{1/2} \tag{6}$$

Step3. To update the winner vectors of the winner node and its neighborhood:

$$w_{jk}(t+1) = w_{jk}(t) + \alpha(t)[x_{ik} - w_{jk}(t)] \quad j \in N(t) \tag{7}$$

$N(t)$ is the non-increasing neighborhood function, $\alpha(t)$ is learning rate function $0 < \alpha(t) < 1$.

Step4. To repeat Step2 and Step3 until KLA learning stabilizes

In this paper, the parameters of KLA used to train the codebook in this paper can be referred in the following Table2.

3.2 Detection

The Vector Quantization framework for the TCP attack detection is described in the following Fig.2. After training, the codebook has been constructed for the normal network traffic. In other words, Voronoi partition [12] is formed in the normal traffic behavior space. Every codeword of codebook represents the centroid of the more similar group according to the minimum Euclidean distances among the input vectors of network traffic. In detection phase, the input feature vectors will be processed to the codebook, which is known as searching best matching codeword, i.e. the nearest neighbor searching. The anchor point named as the nearest neighbor codeword has been found for each input feature vector by exhaustively searching to compute its minimum Euclidean distance to all of the codewords in the codebook. The **Quantization Errors** [2], the Euclidean distance to the nearest neighbor, can be utilized to measure the similarity of short-term behavior in input space and long-term behavior embedded in codebook.

4 Data Processing

Before AID process, it is necessary to do data preprocessing to extract the feature attributes from IP packets, and then, the Date Normalization will be processed to project whole feature attributes to a unit range no matter the continue attributes or quantitative attributes. In the paper, data preprocessing is focused on TCP traffic.

4.1 TCP Flow Feature Attribute

The extraction of feature attributes of network traffic is the foundation of machine learning algorithms in AID. Moreover, excellent detection models or algorithms must be combined with the rational feature vector extraction to improve the attack recognition capability. Traffic features should prefer to differentiate usual traffic profiles from anomaly traffic profiles. The aim of feature extraction is to achieve the maximum difference degree between usual usage behaviors and anomaly behaviors. A feature vector of the TCP traffic flow is shown in Table 3.

Table 3. Feature attributes of TCP Flow feature vector

Feature Attribute	Describe
SrcIP	source IP address
DestIP	destination IP address
SrcPort	source port
DestPort	destination port
PktSize	average packet size in one TCP Flow
SrcBytes	the number of bytes from source
DestBytes	the number of bytes from destination
FlowState	TCP Flow closed state
Fre_SrcIP	frequency of a certain source IP in time-window
Fre_DestIP	frequency of a certain destination IP in time-window

There are nine flags involved in the connection establishment of TCP 3-way handshake protocol and the connection close of TCP 4-way handshake protocol. We devised a 9-bit number to identify the connection state. The flag will be set to 1 if the corresponding flag is observed during the establishment-close process. Otherwise, the corresponding flag will set to 0. A decimal function with the non-superposed value is used to quantitate the whole connection process — $Sum(Flag_0, Flag_1, \dots, Flag_8)$:

$$\begin{aligned}
 FlowState &= Sum(Flag_0, Flag_1, \dots, Flag_8) = \sum_{i=0}^8 Flag_i \cdot 2^i \\
 &= RST_{active} \cdot 2^0 + RST_{passive} \cdot 2^1 + SYN \cdot 2^2 + ACK_{syn} \cdot 2^3 + ACK \cdot 2^4 \\
 &\quad + FIN \cdot 2^5 + ACK_{fin} \cdot 2^6 + FIN' \cdot 2^7 + ACK'_{fin} \cdot 2^8
 \end{aligned} \tag{8}$$

The TCP connection with the normal close is:

$$Sum = (111111100)_2 = (508)_{10}$$

For example: There are two statuses according to one SYN probing attack

- (1) The target port is not open and the target computer responses a $RST_{passive}$:

$$Sum = RST_{passive} \bullet 2^1 + SYN \bullet 2^2 = 1 \times 2^1 + 1 \times 2^2 = 6$$

- (2) The target port is open and the target computer responses an ACK_{syn} :

$$Sum = SYN \bullet 2^2 + ACK_{syn} \bullet 2^3 + RST_{active} \bullet 2^0 = 1 \times 2^2 + 1 \times 2^3 + 1 = 13$$

The number 6 and 13 describe two abnormal TCP connection states in the process of SYN probing attack. Consider the situation that TCP protocol has the re-transmission mechanism and a TCP Flow aggregates some TCP connections so that the certain flag will repeatedly in a TCP Flow, we substituted the 32-bit number (4 bytes) for the 9-bit number, as in Fig 3. So if the occurrence time of one certain flag is less than 15, the sum will not be repeated. The number of occurrence time will take value of 15 if it exceeds 15. ($RST_{passive}/RST_{active}$ occurrence time is less than 3).

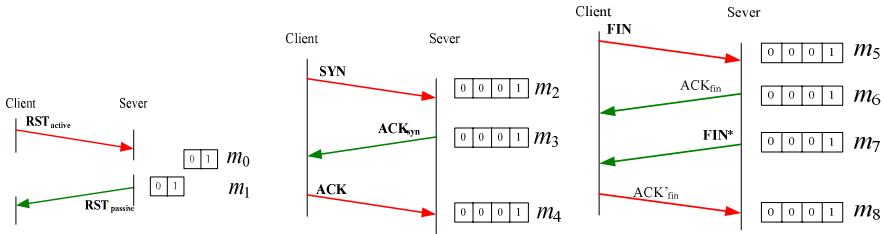


Fig. 3. Quantization of TCP Flow state

The reason that RST takes low-bit and ACK'_{fin} takes high-bit is to maximize the discrepancy between the normal connection closed (ACK_{fin}) and the connection rejected abnormally (RST). The large discrepancy between normal feature attribute and abnormal can help to advance AID accuracy.

The TCP Flow state is the most important feature attribute. Many attacks can result in the abnormal state of connection according to the TCP protocol. Generally, 14 connection-closing states are summarized in AID model based on the data mining [4, 5]. We found that the method of tracking 14 connection states is clumsy relatively in data preprocessing program. What is more important, these 14 states cannot include all the complicated instances of TCP connection state. By state quantization, every state of connection can be mapped to an int data range affording the state synopsis attribute directly leading to the whole improvement of feature vector.

4.2 Time Window

In order to get the correlation and statistical information in a certain time interval (2 seconds in this paper), a time window was designed as in Fig.4. The dashed denotes the flow began out of time window but ended in it while the real line denotes the Whole circle of flow is in the time window. In the data preprocessing the flows closed in time window are counted no matter when did it began which is convenient to the traffic statistical correlation and the data preprocessing.

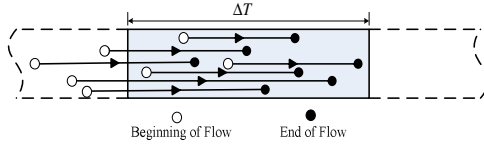


Fig. 4. Time window

5 Evaluation Method and Result

5.1 Experiment Data Set

We take a part of the 1999 DARPA Intrusion Detection Evaluation Data Sets of [13, 14] to estimate the structural Vector Quantization for AID off line. The codebook of

Table 4. Attacks in evaluation

Back (DoS)	3	Ntinfoscan (Probing)	3
Selfping (Probing)	3	Apache2 (DoS)	3
Portswep (Probing)	12	Queso (Probing)	3
Satan (Probing)	2	Neptune/SYN-flood (DoS)	3
Mscan (Probing)	1	Processtable (DoS)	2
Total	35		

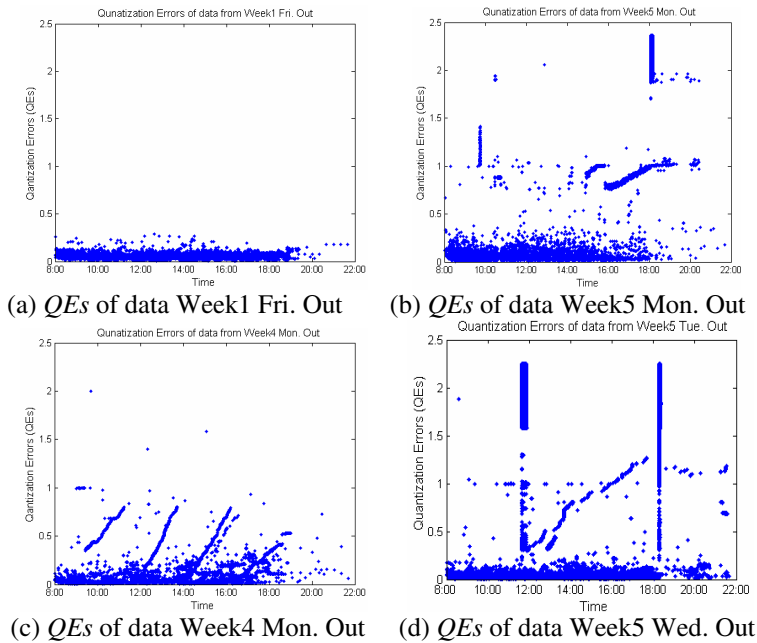


Fig. 5. QE distributions of the training data and testing data

Table 5. Attack detection rates for Week4 and Week5 attacks

Date	Attack ID	Attack Name	DR	QE	FPR
Week4 Mon	41.091531	PortswEEP	1/3(33.3%)	1.3999	0.31%
	41.111531	PortswEEP	5/5(100%)	0.9072~1.2328	
	41.122222	PortswEEP	4/5(80%)	1.1001~1.3125	
	41.162715	PortswEEP	0/10(0%)	0.6723~0.7213	
Week4 Wed	43.080401	Satan	7/16(43.8%)	0.7164~1.3196	0.11%
	43.164334	PortswEEP	3/3 (100%)	1.1281~1.2947	
Week4 Thu	44.080000	Ntinfoscan	9/15 (60%)	0.8793~0.9539	0.14%
Week4 Fri	45.111010	PortswEEP	0/8 (0%)	0.7234~0.7379	0.96%
	45.181011	PortswEEP	5/5 (100%)	0.9012~1.2318	
Week5 Mon	51.084334	PortswEEP	3/3 (100%)	1.0417~1.2256	3.15%
	51.094334	PortswEEP	90/106 (84.9%)	0.8039~1.3017	
	51.102700	Apache2	948/1014 (93.5%)	0.8769~0.8972	
	51.140100	Apache2	11/12 (91.7%)	0.8592~0.8874	
	51.180445	Neptune	19731/20480 (96.3%)	1.8038~2.3063	
	51.201715	Selfping	1/1 (100%)	1.0218	
Week5 Tue	52.094514	Selfping	0/1 (0%)	0.7288	3.25%
	52.103409	Back	47/47(100%)	0.8254~2.0578	
	52.113855	Neptune	40950/40960 (99.9%)	1.9088~2.3583	
	52.165435	Queso	7/7 (100%)	1.0017~1.0021	
	52.181637	Neptune	950/1024 (92.7%)	1.9075~2.3572	
	53.045454	Selfping	5/5 (100%)	0.9059~1.0134	
Week5 Wed	53.102617	Back	40/40 (100%)	0.8906~0.8941	0.37%
	53.110516	Queso	4/7 (57.1%)	1.0218~1.0431	
	53.123735	PortswEEP	8/13 (61.5%)	1.0047~1.2766	
	53.134015	Queso	4/7 (57.1%)	0.8084~0.8507	
	53.150110	Processtable	124/375(33.1%)	0.8821~0.8927	
	53.152648	Back	0/40 (0%)	0.5679~0.7221	
	53.171350	Apache2	155/340(45.6%)	0.8832~0.8959	
	53.195130	PortswEEP	100/100 (100%)	1.0413~1.6032	
Week5 Thu	54.103459	PortswEEP	3/3 (100%)	1.0585~1.3623	0.67%
	54.110416	Ntinfoscan	5/16 (100%)	0.8243~0.8448	
	54.145832	Satan	8817/9120 (96.7%)	0.8023~1.9823	
	54.183002	Ntinfoscan	6/17 (37.3%)	0.8121~0.8528	
	54.195951	Mscan	5724/5724 (100%)	1.0071~1.4995	

(The threshold of *QE* is 0.8)

VQ is designed on the data set attack free in week 3 and week 1. However, consider the fact that the target of this paper is network work attack based on TCP and not general, we test for ten TCP attacks (DoS and Probing mainly), 35 instance attacks total, as showed in Table 4. We filter out some other attacks out of the test traffic data according to the attack identification [13] of 1999 DARPA deliberately after the feature vectors extracted. A detailed description of these attacks could be found in [13].

5.2 Quantization Errors and Result

We use *Quantization Errors (QEs)* to evaluate the on-detecting network traffic vectors. Fig.5 (a-d) presents the *QEs* distributions in outside traffic of 4 days in DARPA data set including one the training data Week1 Fri. and the three-day testing data. The DARPA data of week1 Fri. begins in 8:00 morning and ends about in 22:08 evening (22:04 about in week5 Mon.) after which few data can be observed in those two days. [13]

From the Fig.5 (a), we can observe that *QEs* in training data is well regulated and don't change with the large deviation. However, the strong contrast in Fig.5 (b) is the sharp variation of *QEs* due to the high values of attack traffic *QEs*. Markedly, on 18:04, Neptune attack (Syn-flood) can be viewed with *QEs* values ranging from 1.8038 to 2.3063 in Fig.5 (b). The same case happens in other day data which just can be viewed by Fig.5 (c) and Fig.5 (d).

The overall outcome of evaluation is present in Table 5. The Detection Rate (DR) and False Positive Rate (FPR) are both presented for the whole day circumstances accordingly. Individuals in one attack circumstance are given singly for the better understand. Obviously, every attack with certain ID is composed of the attack multi-flows, from a few to the huge volume such as Neptune because many DoS/DDoS and Probing attacks behave the style of the bursty network traffic. These bursty attacks usually manoeuvre the huge of network traffic to attack computer servers. The single or a few of TCP flows usually are used to explore the service information of some servers, for instance, if the port of 8080 is open in servers. We can conclude that our intrusion detection method exhibit more robust to these bursty attacks with the huge multi-flows and get the higher DR with the lower FPR.

6 Conclusions

The paper proposed an intrusion detection mechanism using the structural Vector Quantization, especially to detect DoS/DDoS and Probing attacks for web services. With the codebook, the normal usage profile of the temporal-spatial scale network traffic could be constructed quantitatively to describe the normal long-term behaviors. The evaluation experiments confirmed that our anomaly intrusion detection framework can achieve the higher detection rate with the lower false detection rate.

References

1. Moore D., Voelker G., and Savage S., :Inferring Internet Denial-of-Service Activity, in Usenix Security Symposium, Washington, D.C., (2001) 401-414
2. Robert Gray and David L. Neuhoff: Quantization. IEEE Transactions on Information Theory, Vol. 44, (1998) 2325-2384

3. D. E. Denning: An Intrusion-detection Model. *IEEE Transactions on Software Engineering*, Vol. 13(2), (1987) 222-232
4. E. Eskin, A. Arnold, M. Prerau: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. *Applications of Data Mining in Computer Security*, Kluwer, 2002
5. Andrew H. Sung, Srinivas Mukkamala: Identifying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks. *Proceedings of the 2003 Symposium on Applications and the Internet*, (2003) 119-123
6. Y. Qiao, X. W. Xin, Y. Bin and S. Ge: Anomaly Intrusion Detection Method Based on HMM, *Electronics Letters*, 38(13), (2002) 663-664
7. J. M. Bonifacio, E. S. Moreira: An Adaptive Intrusion Detection System Using Neural Network, *Research Report*, UNESP, Brazil, 1997
8. <http://www.snort.org>
9. Linde, Y., Buzo, A. and Gray, R. M.: An Algorithm for Vector Quantizer Design. *IEEE Transactions on Communications*, 28(1), (1980) 84-95
10. Ueda, N. and Nakano, R.: A New Competitive Learning Approach Based on an Equidistortion Principle for Designing Optimal Vector Quantizers. *IEEE Transactions on Neural Networks*, 7(8), (1994) 1211-1227
11. Kohonen, T.: *Self-Organization Maps*, 3rd ed., Springer-Verlag, Berlin, 1997
12. Tom Mitchell: *Machine Learning*, McGraw Hill, New York, 1997
13. <http://www.ll.mit.edu/IST/ideval/index.html>
14. Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, Kumar Das.: The 1999 DARPA Off-Line Intrusion Detection Evaluation, *Computer Networks*, 34 (4), (2000) 579-595

cGridex: Efficient Processing of Continuous Range Queries over Moving Objects

Xiaoyuan Wang, Qing Zhang, Weiwei Sun, Wei Wang, and Baile Shi

Department of Computing and Information Technology,
Fudan University, Shanghai, China
{xy_wang, wwsun, weiwang1, bshi}@fudan.edu.cn
qzhang79@yahoo.com

Abstract. In mobile computing environments, the continuous range query is one of the most important types of queries required to support various location-based services. With a large number of queries, the real-time response to query answers and the concurrent execution are two major challenges. In this paper, we propose a novel approach cGridex for efficient processing of continuous range queries over moving objects. cGridex abstracts the dynamic relations between moving objects and concurrent queries, and introduces the concept of coloring on a high level. With colored grids, it forms a direct mapping between moving objects and continuous queries, and can process both of them in a unified way, without redundant evaluation. By precomputing and materializing the color transferring relations, it avoids frequent operations between query sets and spatial join between objects and queries at each time interval, thus leading to fast query evaluation. The experimental results show that cGridex has good performance for the evaluation of continuous range queries.

1 Introduction

With the rapid advances of wireless communications and electronic technologies, Location Based Services (LBS), which track the location of mobile users and provide useful information associated with locations, is becoming a new and important application area. Efficient query processing for moving objects can substantially improve the quality of the services and there arises the great need of new query processing technologies in mobile environments.

The continuous range query is one of the most important type of queries required to support many of the services, such as intelligent transportation systems and sensor-based monitoring. With various applications, a large number of continuous range queries are repeatedly evaluated in a concurrent execution environment. One challenge is the real-time response to query answers, which is critical for actual applications. Any delay of the query response results in an obsolete answer. Examples are traffic monitoring and enhanced 911 services. Another challenge is the concurrent execution. With a large quantity of queries, many query regions interact and lead to overlapping. Movement of a single object can affect the answers of multiple queries, which increases the complexity of evaluation and workload of the server.

Incremental evaluation and query index have been proposed to be two important approaches for continuous range queries [7,8]. To the best of our knowledge, the SINA [7] is the only algorithm that exploits both of these two paradigms. The existing incremental approaches are limited in three aspects: (1)Redundant processing. (2)Query overlapping. (3)Frequent query set operations.

In this paper, we propose a novel approach cGridex (*Colored Grid Based Index*) for efficient processing of concurrent continuous range queries over moving objects. cGridex is designed with two distinguishing features: (1) Enhanced incremental evaluation. (2) Enabling moving objects query-aware. Enhanced incremental evaluation entails that only queries whose answers are changed and only objects that change query answers are wholly processed while others are excluded from answers at an early time. cGridex eliminates redundant processing of both objects and queries in order to minimize computational costs. The query-aware ability is that an object can directly obtain incremental query answers without additional overhead when it moves to a new position.

cGridex abstracts the dynamic relations between moving objects and concurrent queries, and introduces the concept of coloring on a high level. It colors up the grid space and utilizes the space characteristic of grid-based index by dividing a range query into normalized parts and fragmented parts to optimize disk I/O and CPU time respectively. With colored grids, it forms a direct mapping between moving objects and continuous queries, and can process both of them in a unified way, without redundant evaluation. By precomputing and materializing the color transferring relations, it avoids frequent operations between query sets and spatial join between objects and queries at each time interval, thus saving both I/O and computational costs.

The rest of the paper is organized as follows. Section 2 describes the system model and reviews some related work. Section 3 presents our approach cGridex, including its basic idea, data structures, and query algorithms. Section 4 conducts the experimental evaluation and we conclude the paper in Section 5.

2 Preliminary

2.1 System Model

In this subsection, we present the system model and identify the major problems. In concurrent continuous query execution environments, we divide time interval into two kinds: one for query evaluation (ΔT_q), which is equal to the minimum time unit, and the other for location update of moving objects (ΔT_u). ΔT_u is larger than or equal to ΔT_q , which indicates that cycles of location update and query evaluation are not necessarily synchronized. At each ΔT_u , location updates with no client overhead are issued by positioning systems such as GPS or sensor networks. We do not make any assumption about moving objects's storage or computational capabilities. Therefore, only minimal possible information, including identifier and new location of moving objects, needs to be reported to servers.

To investigate how movements of objects influence query answers, we consider the process from two aspects. In a time interval, in the view of a moving object o , there are three cases that may occur: ($o1$) o does not issue a location update.



Fig. 1. Dynamic Relations between Objects and Queries

(o2) o reports the new position to the server, and new pos is equal to old pos.
 (o3) o reports the new position to the server, but new pos is different from old pos. In the view of a query q , two cases will occur: (q1) q 's result remains unchanged. (q2) q 's result is changed. The relation between these two categories is illustrated in Figure 1(a). All the three cases in the object category can lead to case q1 while q2 only comes from o3.

With a large number of queries, the high demand for query responses, and the concurrent execution, there arise the following limited aspects in existing incremental approaches:

- (1) Redundant processing. In general incremental evaluation, we have to carry out the same processing for *all* the objects that change their locations, even for those that do not change query answers. For example, in Figure 1(b), when an object o moves to a new position p'' , the same processing like p' should be performed before we know that o does not change the query result and finally exclude it from the incremental answer. With a large number of objects, such unnecessary processing degrades performance.
- (2) Query overlapping. Since different queries may be applied for different applications, concurrent execution of range queries results in the continuous overlapping of query regions, which can cover a lot of areas. Query overlapping indicates that an object may enter/leave *multiple* query result sets in a positive/negative update. This increases the length of query lists maintained for an index as well as evaluation complexity in a dynamic environment, since movement of a single object can influence answers of multiple queries.
- (3) Frequent query set operations. For incremental evaluation, we have to 1) maintain the queries that objects satisfied at last time interval, 2) execute set operations such as intersection or difference between the old and new query sets that objects satisfy in order to determine positive and negative update answers. For example, in the SINA, for an object, one intersection operation and two difference operations of query sets should be executed. In a large-scale concurrent execution environment, frequently repeated set operations incur heavy CPU costs.

2.2 Related Work

Unlike traditional snapshot queries, continuous range queries have the following characteristics, which have been described in previous work [7,8]: (1) Incremental

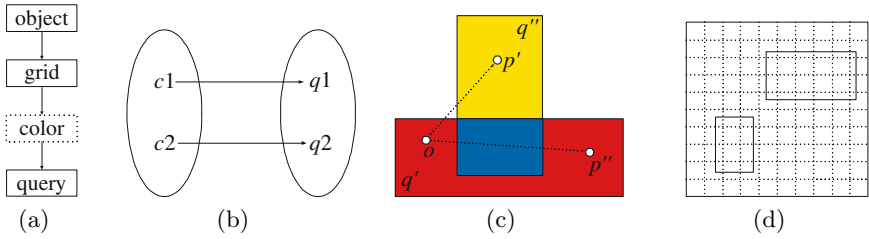


Fig. 2. The Basic Idea

evaluation is more efficient than non-incremental approaches. It is important to maintain query answers continuously rather than to rebuild the whole answers each time. Reevaluating all the queries is costly and unnecessary. With incremental approaches, only the change of query answers needs to be computed, which leads to fast evaluation as well as smaller answer sizes. (2) Query index performs better than the index built on moving objects, since building an index on queries avoids the high cost of keeping an object index updated.

Evaluation of continuous range queries over moving objects has been investigated recently. One way is to directly evaluate the queries on an object index [6,9] at each time, where frequent updates of the object index is a major problem. [7] proposes the SINA, an incremental hash-based algorithm for continuous spatio-temporal queries. It exploits two main paradigms: shared execution and incremental evaluation. SINA distinguish two kinds of query updates: *positive updates* and *negative updates*. Positive/Negative updates indicate that a certain object needs to be added to/removed from the result set of a certain query. In [8], Q-index is presented, which is to build an R-tree-like index structure on queries instead of objects. In order to avoid excessive location updates, a safe region for each moving objects is defined. [5] investigates the main memory evaluation of continuous range queries. It shows that a grid-based query index performs better than an R-tree based query index. [3] proposes a motion adaptive indexing schema for moving continual queries. It indexes less frequently changing objects, which helps decrease the number of updates to the index. Different from the system model in this paper, some previous work [1,2] assumes that moving objects have computational and storage capabilities to share the workload of query processing with the server, which may be not always realistic in actual scenarios.

Some other types of continuous spatial queries have also been investigated, such as continuous k NN queries [12], and spatial semijoin queries [4].

3 cGridex

3.1 Motivation

The main idea behind cGridex is to abstract the dynamic relations between movements of objects and query answers by coloring up the grid space. In the procedure of evaluation, we add a color level, and treat it as a high-level abstract of queries, as is shown in Figure 2(a).

With colored grids, we have another way to view an object's movement: (c1) o is in the same color. (c2) o is in a different color. Therefore we can form a one-to-one mapping between objects and queries, as is shown in Figure 2(b), which is in a unified way and essentially different from that in Figure 1(a). We reconsider the case in Figure 1(b). Different query regions, including overlapping regions, take on different colors, as is shown in Figure 2(c). Then we can easily identify whether an object incurs any change of query answers by its color. Further, if we materialize the relations between colors and queries, in continuous execution, we can directly obtain query answers without extra computational costs. With such a basic idea, we could have the following algorithm sketch of query evaluation:

Algorithm Sketch

1. identify an object's current color.
 2. obtain the incremental answers directly from the materialized relations by the new and old colors.
 3. update the object information.
-

3.2 Data Structure

The space is equally divided into $N \times N$ small grids. A grid, query, and color are respectively identified by GID , QID , and CID . Specifically, CID is 1-based and 0 is reserved for no color.

First we describe the preprocessing before index construction, which includes three steps:

1. Query Normalization. A query q is projected on the grid space and is divided into the normalized part and fragmented part. The normalized part of q is the query whose region is made up of all the grids fully covered by q . The rest of q is the fragmented part, as is shown in Figure 2(d).
2. Coloring of query regions. First we project the normalized query regions on the grid space, and color up them. Then we clear the grid space, project the fragmented query regions on it, and color up them. Both of the procedures obey the *coloring rule*: (1) Regions covered by different queries take on different colors. (2) Two regions take on the same color if and only if they are covered by the same queries. (3) All the regions that are not covered by any query are colored 0.
3. Color selection for grids. Each grid takes on two kinds of colors: full color and partial color. A grid has only one full color and can have multiple partial colors. If the grid is fully covered by the query region in color fc , its full color is fc . If the grid is partially covered by the query region in color pc , pc becomes one of its partial colors. In other words, colors that come from normalized/fragmented query regions are the full/partial colors. For simplicity, we use $R(q)$, $NR(q)$, and $FR(q)$ to denote the region of query q , the normalized query region of q , and the fragmented query region of q . We introduce the following definition first.

Definition 1. (Relation *contains*) A query q contains a color c if some part of q 's region takes on color c .

During the course of its execution, cGridex maintains the following data structures:

- **Colored Grid Based Index (CGI).** CGI is an in-memory sequential structure indexed on GID . With row-major ordering, each grid in the space has one entry in CGI. An index entry has the form (FC, ptr) , where FC is the full color and ptr is a pointer to the partial color list $PCList$, which is stored on disk. Each entry in $PCList$ has the form (PC, QR) , where QR is the corresponding partial query region in the grid and PC is the partial color that QR takes on.
- **Color-Query Table (CQT).** CQT is a disk-based sequential table indexed on colors. Each entry in CQT has the form $(C, QList)$, where $QList$ is a list that includes all the queries that contain color C . In $QList$ queries are identified by QID . In CQT, we do not distinguish full colors and partial colors, which indicates that full colors and partial colors are unified from the view of queries.
- **Hash Table for Color Pairs (HT).** HT is an in-memory hash table, with its hash values on disk. The key of HT has the form (c_i, c_j) , where color c_i and c_j satisfy the color transferring relation, which will be described in Subsection 3.3. The value of HT is a pointer to the materialized relations on disk.
- **Object Table (OT).** OT is a disk-based table for storing object information. An entry in OT has the form (OID, FC, PC) , where OID is object identifier, and FC and PC are the full color and partial color that the object is in at the last time.
- **Query Table (QT).** QT stores the query information, including QID and the query region. It is used only when there are queries to be inserted or deleted.

3.3 Materialized Relation

In this subsection, we describe how to construct and materialize the color transferring relations, which could speed up query processing in real-time environments. First we introduce two definitions.

Definition 2. (Query Overlapping Relation, *QOR*) Query q_i and q_j satisfy *QOR* if and only if $R(q_i)$ overlaps with $R(q_j)$.

Definition 3. (Color Transferring Relation, *CTR*) Color c_i and c_j satisfy *CTR* if and only if (1) Both c_i and c_j are full colors or partial colors. (2) $\exists q_i \in CQT[c_i].QList, \exists q_j \in CQT[c_j].QList$, satisfy $q_i = q_j$.

Color Transferring Relation reflects that there are common queries in the $QList$ of color c_i and c_j , which indicates that if an object moves from c_i to c_j ,

query set operations are needed in order to get incremental answers. Algorithm 1 describes how to generate the *CTR* for all the colors. We maintain a temporary array to record all the colors that a query *contains*. It should be noted that step 1 ~ 3 in the algorithm can be simultaneously done in the preprocessing step 2 in Subsection 3.2.

Algorithm 1. Generate *CTR*

1. **if** a new color c is generated
 2. **for** each query k that contains c
 3. add c to array[k].
 4. **for** each query k
 5. **for** each $c_i, c_j \in \text{array}[k]$, and $c_i < c_j$
 6. **if** pair(c_i, c_j) has not been added to HT
 7. add (pair(c_i, c_j), ptrResult(c_i, c_j)) to HT.
-

ptrResult is a pointer to the results of $CTR(c_i, c_j)$, which are materialized on disk. The result has the form (L_i, L_j) , where $L_i = \text{CQT}[c_i].QList - \text{CQT}[c_j].QList$, and $L_j = \text{CQT}[c_j].QList - \text{CQT}[c_i].QList$. With the materialized *CTR*, positive answers (PA) and negative answers (NA) can be obtained directly. Since dynamic transferring relations have been precomputed, there is no need to execute set operations such as intersection or difference in dynamic real-time environments.

Algorithm 2. GetIncrementalAnswers(color i , color j)

1. $p = \text{HT}(i, j)$.
 2. **if** $p \neq \text{NULL}$
 3. obtain the materialized result (L_i, L_j) on disk by p .
 4. positive = L_j .
 5. negative = L_i .
 6. **else**
 7. positive = $\text{CQT}[j].QList$.
 8. negative = $\text{CQT}[i].QList$.
 9. **return** (positive, negative).
-

Algorithm 2 describes how to obtain incremental answers with the materialized *CTR*, which is identical for both full colors and partial colors. Without loss of generality, an object moves from color i to color j and $i < j$. In the algorithm, if color i and j satisfy the *CTR*, we can get the materialized incremental answers from HT. Otherwise the answers can be got from the corresponding *QList*. With the implementation of dynamic arrays for the *QList*, the algorithm

complexity is $O(1)$. An example is shown in Figure 3. $CQT[c1].QList = \{q1, q2, q3\}$, $CQT[c2].QList = \{q3, q4, q5\}$, $CQT[c3].QList = \{q5, q6\}$. Then

$$HT(c1, c2) = \begin{cases} \{q1, q2\} \\ \{q4, q5\} \end{cases}$$

If object o moves to $p1$, according to $HT(c1, c2)$, PA is $\{q4, q5\}$, and NA is $\{q1, q2\}$. If o moves to $p2$, $PA = CQT[c3].QList$ and $NA = CQT[c1].QList$, since color $c1$ and $c3$ do not satisfy the relation CTR .

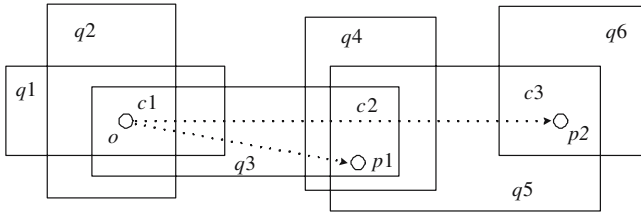


Fig. 3. Example to Show Materialized Incremental Answers

3.4 Query Algorithm

According to the coloring rule in Subsection 3.2, a region takes on two kinds of colors: full color and partial color. We use the form $(fc1, pc1)$ and $(fc2, pc2)$ to denote the corresponding full color and partial color an object takes on in the old and new position. An object identifies its full color and partial color at each query time. We introduce two lemmas below, which guarantee the correctness of the query evaluation. Details of proof can be found in [11].

Lemma 1. Given a position p takes on the color pair (fc, pc) , there does not exist such a query q that satisfies both $q \in CQT[fc].QList$ and $q \in CQT[pc].QList$.

Lemma 2. Given an object's old color $(fc1, pc1)$ and new color $(fc2, pc2)$, there is no duplicate in the incremental answers that result from $fc1 \rightarrow fc2$ and $pc1 \rightarrow pc2$ respectively.

Although the full color and partial color are processed separately in the evaluation and there is not duplicate in both of the answers according to Lemma 2, there are some cross influences on each other, which are verified in Algorithm 3. The cross validation phase is to remove the unnecessary incremental answers.

Before the evaluation an object should identify its current color, and Algorithm 4 describes this procedure. For the full color, we can directly obtain it from CGI. For the partial color, we get the $PCList$ on disk and checking whether the new position lies in the corresponding partial query region.

Algorithm 3. CrossValidation($fc1, pc1, fc2, pc2$)

1. **if** $fc1 \neq fc2$
 2. **if** $pc1 = fc2$
 3. remove the positive answer of $fc1 \rightarrow fc2$.
 4. **if** $fc1 = pc2$
 5. remove the negative answer of $fc1 \rightarrow fc2$.
 6. **if** $pc1 \neq pc2$
 7. **if** $fc1 = pc2$
 8. remove the positive answer of $pc1 \rightarrow pc2$.
 9. **if** $pc1 = fc2$
 10. remove the negative answer of $pc1 \rightarrow pc2$.
-

Algorithm 4. IdentifyColor(pos)

1. $i = pos.x / xunit, j = pos.y / yunit$.
 2. $k = j \times N + i$.
 3. $fc2 = CGI[k].FC$.
 4. $pc2 = 0$.
 5. **for** each $(PC, QR) \in CGI[k].PCList$
 6. **if** pos is within QR
 7. $pc2 = PC$.
 8. **break**.
 9. **return** $(fc2, pc2)$.
-

Algorithm 5. GetAnswers($fc1, pc1, fc2, pc2$)

1. set $FPos, FNeg, PPos, PNeg$ to NULL.
 2. **if** $fc1 \neq fc2$
 3. $(FPos, FNeg) = \mathbf{GetIncrementalAnswers}(fc1, fc2)$.
 4. **if** $pc1 \neq pc2$
 5. $(PPos, PNeg) = \mathbf{GetIncrementalAnswers}(pc1, pc2)$.
 6. **CrossValidation**($fc1, pc1, fc2, pc2$).
 7. **return** $(FPos \cup PPos, FNeg \cup PNeg)$.
-

It should be noted that, different from normalized queries, some fragmented query regions can not be represented by a rectangle. We use an ordering technique to solve this problem. We make such a query stored behind the queries that are fully covered by it in the $PCList$.

Algorithm 5 describes the whole procedure of getting incremental answers. The results consist of two parts, positive answers and negative answers. The algorithm proceeds by the full color and partial color respectively, and combines the two kinds of answers after the cross validation phrase. Now we extend the algorithm sketch in Subsection 3.1 to Algorithm 6, which describes the query evaluation in cGridex.

Algorithm 6. QueryEvaluation(*obj*, *pos*)

1. obtain old color (*fc1*, *pc1*) from OT[*obj*].
 2. obtain new color (*fc2*, *pc2*) by **IdentifyColor**(*pos*).
 3. obtain positive and negative answers by **GetAnswers**(*fc1*, *pc1*, *fc2*, *pc2*).
 4. **if** (*fc1*, *pc1*) \neq (*fc2*, *pc2*)
 5. update OT with (*fc2*, *pc2*).
 6. **return** answers.
-

Further, details of the index maintenance and the performance analysis of I/O and CPU cost, as well as memory requirement, can be found in [11].

4 Experimental Evaluation

4.1 Experimental Setup

In the experiments, we compare the performance of the following three approaches: cGridex, SINA and Q-Index. We implement the algorithms using C++. All the experiments are conducted on Pentium IV 2.2GHz with 512 MB RAM running Windows Server 2003. The size of disk page is set to 4KB. We use synthetic datasets generated by the "General_Spatio_Temporal_Data" (GSTD) [10]. All the objects are uniformly distributed in a square unit and can move arbitrarily with a maximum moving distance of 0.2. The default number of objects is 100K and the default number of queries is 10K. The average query size is 0.05. The performance metric is the number of disk access and CPU time, which is considered as the time used to perform computation in memory.

According to the system model, we set ΔT_q to 10 seconds and ΔT_u to 60 seconds, that is, all the queries are evaluated every 10 seconds and each moving objects reports its new location information every 60 seconds. The percentage of moving objects that report a change of location information within ΔT_q is 30%. In cGridex, k is set to 100, which determines the number of grids in the space.

4.2 Results and Discussion

We investigate the effects of the number of moving objects and continuous queries respectively in the experiments.

Figure 4(a) and 4(b) show the effect of increasing the number of moving objects from 10K to 100K on I/O cost and CPU time. In Figure 4(a), both SINA

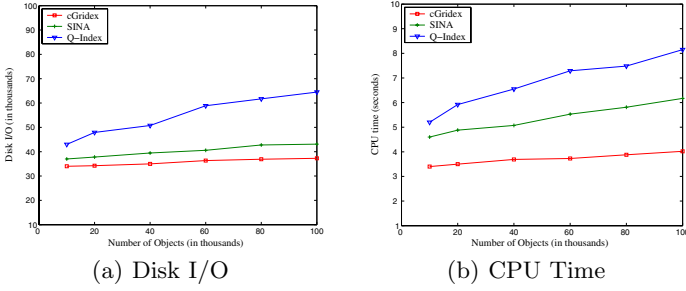


Fig. 4. Varying Number of Objects

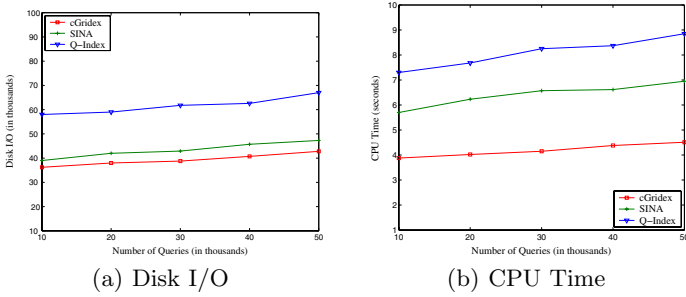


Fig. 5. Varying Number of Queries

and cGridex shows its scalability. With the increase in the number of moving objects, the performance of SINA and cGridex keeps stable. Both of them are the incremental evaluation, and only the change of query answers needs to be computed, avoiding the repeated reevaluation of queries. However, the performance of Q-Index degraded, because it reevaluates all the objects and rebuilds the query answers at each query evaluation time. It should be noted that both SINA and cGridex employ the indexing of objects (the Object Index in SINA and the Object Table in cGridex), which provides the support for the incremental evaluation, while objects are not indexed in Q-Index. We also see that cGridex performs slightly better than SINA, because with the enhanced incremental evaluation only objects that change query answers proceed to be wholly processed while others are excluded from answers at an early time, which eliminates the redundant processing. In Figure 4(b), cGridex has the lowest CPU time. The reason is that the dynamic transferring relation between queries and moving objects has been precomputed and materialized, and therefore the incremental answers can be obtained directly without additional computational overhead. As is shown in the figure, SINA computes the answers dynamically each time, and the cost slightly increases with the increase of the number of objects.

Figure 5(a) and 5(b) give the effect of the variable number of continuous queries, which increases from 10K to 50K. In Figure 5(a), we see that Q-Index has the high I/O cost since it is based on the R-tree structure indexed on queries and non-indexed objects worsen the performance. Both of SINA and cGridex

increase its I/O cost slightly, because with the more queries to be evaluated, the structures maintained for queries, the Grid Index in SINA and the Color-Query Table in cGridex, need more disk space, which lower the performance of one disk access. In Figure 5(b), when the number of queries becomes larger, SINA increases its CPU cost since it repeatedly executes the set operations such as difference and intersection between old and new query lists, whose length have increasingly become longer. With the materialized coloring transferring relations, cGridex is not much affected by the increase of queries.

5 Conclusion

Efficient evaluation of continuous range queries has increasingly been important in a wide range of application areas. In this paper, we propose a novel approach, namely cGridex, for efficient processing of continuous range queries over moving objects. cGridex abstracts the dynamic relations between moving objects and concurrent queries, which aims at avoiding redundant evaluation. It precomputes and materializes the color transferring relations, without dynamic frequent computation. In the future work, we will further extend the basic idea to other types of continuous queries over moving objects.

References

1. Y. Cai, and K.A. Hua. Processing Range-Monitoring Queries on Heterogeneous Mobile Objects. In *Proc of MDM*, 2004.
2. B. Gedik, and L. Liu. MobiEyes: Distributed Processing of Continuously Moving Queries on Moving Objects in a Mobile System. In *Proc of EDBT*, 2004.
3. B. Gedik, K.-L. Wu, P. Yu, and L. Liu. Motion Adaptive Indexing for Moving Continual Queries over Moving Objects. In *Proc of CIKM*, 2004.
4. G. S. Iwerks, H. Samet, and K. Smith. Maintenance of Spatial Semijoin Queries on Moving Points. In *Proc of VLDB*, 2004.
5. D. V. Kalashnikov, S. Prabhakar, S. E. Hambrusch, W. G. Aref. Efficient Evaluation of Continuous Range Queries on Moving Objects In *Proc of DEXA*, 2002.
6. M.L. Lee, W. Hsu, C.S. Jensen, B. Cui, K. L. Teo. Supporting Frequent Updates in R-trees: A Bottom-Up Approach. In *Proc of VLDB*, 2003.
7. M. F. Mokbel, X. Xiong, and W. G. Aref. SINA: Scalable Incremental Processing of Continuous Queries in Spatio-temporal Databases. In *Proc of SIGMOD*, 2004.
8. S. Prabhakar, Y. Xia, D. V. Kalashnikov, W. G. Aref, and S. E. Hambrusch. Query Indexing and Velocity Constrained Indexing: Scalable Techniques for Continuous Queries on Moving Objects. *IEEE Trans. on Computers*, 51: 1124-1140, Oct. 2002.
9. S. Saltenis, C.S. Jensen, S.T. Leutenegger, and M.A. Lopez. Indexing the Positions of Continuously Moving Objects. In *Proc of SIGMOD*, 2000.
10. Y. Theodoridis, J.R.O. Silva, and M.A. Nascimento. On the Generation of Spatiotemporal Datasets. In *Proc of SSD*, 1999.
11. X. Wang, Q. Zhang, W. Sun, W. Wang, and B. Shi. cGridex: Efficient Processing of Continuous Range Queries over Moving Objects. Technical Report, Fudan University, 2005.
12. X. Xiong, M. F. Mokbel, and W. G. Aref. SEA-CNN: Scalable Processing of Continuous K-Nearest Neighbor Queries in Spatio-temporal Databases. In *Proc of ICDE*, 2005.

An Efficient Context Modeling and Reasoning System in Pervasive Environment: Using Absolute and Relative Context Filtering Technology*

Xin Lin¹, Shanping Li¹, Jian Xu², Wei Shi¹, and Qing Gao¹

¹ College of Computer Science, Zhejiang University, Hangzhou, P.R.China 310027

² College of Computer Science, Hangzhou Dianzi University,
Hangzhou, P.R.China 310000

alexlinxin@hotmail.com, shan@cs.zju.edu.cn,
cnxujian@hotmail.com, shiwei@zj165.com, tsing.gao@eyou.com

Abstract. Challenges revealed in designing efficient context modeling and reasoning systems in pervasive environment are due to the overwhelming contextual information in such environment. In this paper we aim at designing an attribute-based context filtering technology (ACMR) to improve the performance of context processing. Two metrics, absolute and relative attributes, are proposed in our work to analyze the contextual information. ACMR only processes the application-related contextual information rather than all the available contextual information to prevent context-aware applications from being distracted by trashy contexts. Additionally, to encourage the reuse and standardization, contexts ontology TORA is developed to model the contexts and their absolute attributes in the pervasive environment. Experiments about ACMR system demonstrate its higher performance than those of previous systems.

1 Introduction

Invisibility is a key requirement of pervasive computing. To achieve it, applications in pervasive environment have to adapt their behaviors to the context, which is defined as any information that can be used to characterize the situation of the environment [1]. As a result, using contextual information as implicit input is one of the major characteristics of pervasive computing.

In pervasive computing community, much work focuses on context modeling and reasoning, which deal with high-level abstraction and inference of pervasive contextual information. Several prototype systems have been proposed [2] [4], but most of the systems take all available contextual information as the input without classification. The amount of the contextual information is so huge that these systems are inefficient, even useless.

In this paper, an attribute-base context filtering technology (ACMR) is proposed to improve the performance of contextual information processing. We argue that a specific application do not care about all contextual information. It's necessary to

* This work is supported by National Natural Science Foundation of China (No. 60473052).

design a mechanism to evaluate the contextual information. In ACMR, two metrics, absolute and relative attributes, are proposed in our work to analyze the contextual information. The *absolute attribute* of contextual information records the characteristics of the contexts, such as, the age of a context or the range that the context covers. When a piece of context is collected, its absolute attributes are also recorded by context sources. Different applications may have various requirements on the absolute attributes of the contextual information. E.g., a dynamic map loading application, which can react to the owner's location, would require the precision of the location information within 2 meters. The *relative attribute* of contextual information reflects the relevance between the contextual information and context-aware application. An application are not concerned about every piece of context. For example, Tom's mobile phone only cares about the contextual information of Tom not that of Jerry. Base on this idea, context absolute filter and relative filter are designed in ACMR. These filters discard the contextual information whose absolute or relative attributes does not reach the applications' requirements and reduce the overhead of the context reasoner. Additionally, to encourage the reuse and standardization, contexts ontology TORA is developed to model the contexts and their absolute attributes in the pervasive environment. Experiments demonstrate such reduction of the input greatly improves the performance of the context processing system.

"The rest of the paper is organized as follows. The Section 2 compares the ACMR with the related works and TORA is introduced in the Section 3. Section 4 depicts the overview and main data structures of ACMR respectively. Section 5 presents the detail of the attribute-based filtering technology. We evaluate the performance in Section 6 and the conclusion is drawn in Section 7."

2 Related Work

There are several context modeling and reasoning systems. X.H.Wang [2] [3] proposed ontology-based context modeling and reasoning. Anand [4] use probabilistic and fuzzy logic to deal with reasoning about uncertain contexts in pervasive environment, which is also on basis of ontology. However, these works only focus on how to get high-level contexts from raw contexts. They did not care about how to consume the high-level contexts. Moreover, in their reasoning mechanism, all raw contexts are adopted as the input of reasoner, no matter whether they are useful for applications. Such solutions may impose high overhead on the system.

Our previous work has proposed the AOCF [7] technology to deal this problem. It adopts Concerned Value algorithm to analyze the relevance between the applications and contexts. The relevance reflects the relative attributes of the contextual information, while we consider the absolute attributes are also important in this paper. T. Buchholz [9] has proposed the term "QOC" to represent the absolute attributes of contexts. K. Henriksen [10] address absolute attributes of contexts through tagging quality parameters to associations between entities and attributes. However, these works only address the modeling of the absolute attributes and they don't give a concrete method to analyze and utilize them.

3 Context Ontology TORA

As aforementioned, in ACMR system, to encourage the reuse and standardization of contextual information, we define context ontology TORA using OWL [11] (Web Ontology language), which is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language. According to Schilit [8], contexts in pervasive computing can be divided into three categories: user context, computing entity context and physical context. Corresponding to the above three categories respectively, we define three OWL classes, which form the skeleton of TORA. All other classes are sub-class of them. Fig.1 shows a part of TORA. In realistic pervasive computing environments, there are special concepts in each domain. For simplicity, we adopt first order to represent the contextual information in TORA, that is, each piece of contextual information has a subject, a predicate and an object.

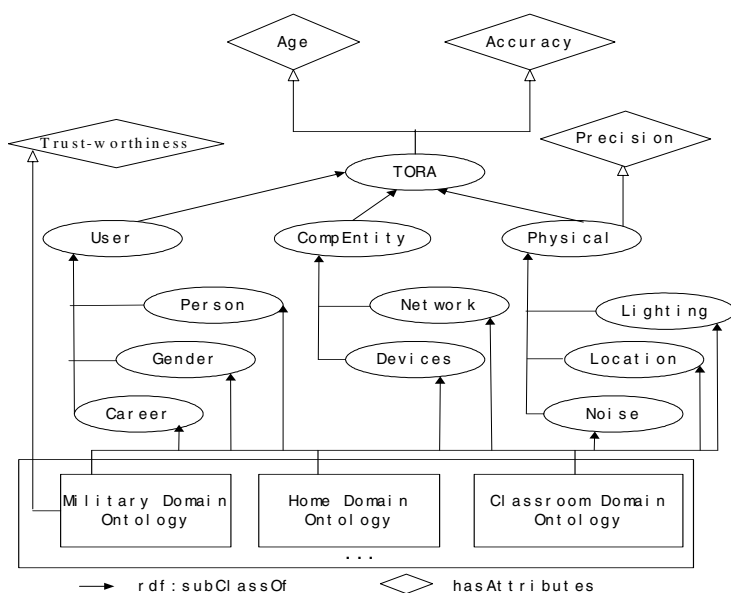


Fig. 1. A part of TORA

Comparing to previous context ontologies [2][3][4], TORA takes absolute attributes of contextual information into account. As shown in Fig.1, all contextual information may have two absolute attributes, age and accuracy. The age indicates the time that since the generation of the contextual information. Since the contextual information may change over time, that a piece of context is sensed earlier means its reliability is lower. In general, age will be specified by adding a time-stamp to context information. Thus, clock synchronization between the context source and the context consumer is needed. Accuracy denotes the probability of that a piece of context is correct. Some of the context providers are unreliable sensors, which may provide the

wrong contextual information. These context providers may estimate the accuracy of the contextual information they provide.

Each subcategory of TORA and domain ontology can define the absolute attributes if need. For example, in military domain, there may be some fake contextual information provided by enemies. So it's necessary to mark the trust-worthiness of the contextual information in military domain. Since the contextual information in the physical subcategory characterizes the situation of physical entities, it is reasonable to mark precision in such contexts, e.g., the precision of temporal contexts is 1 millisecond.

4 Overview of ACMR

4.1 Architecture

Similar to our previous work [7], the ACMR system consists of 5 components: context reasoner, context knowledge base (CKB), context modeler, context filter and context-aware applications (Fig. 2).

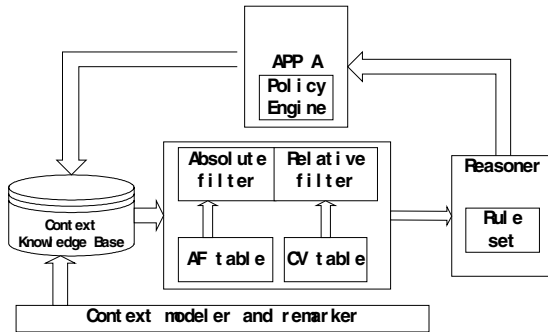


Fig. 2. The Architecture of ACMR

The context modeler is the interface between environmental devices and the ACMR system. After receiving contexts from heterogeneous context sources, the context modeler uses TORA to mark these contexts as well as their absolute attributes. E.g. if a location sensor detects Tom is in RoomA in Mar. 8, 2005 10:00 pm, the context modeler marks this context as Fig.3:

```

<Context rdf:ID = "001253">
  <hasSubject rdf:source="Tom"/>
  <hasPredicate rdf:source="locatedIn"/>
  <hasObject rdf:source="RoomA"/>
  <hasTimeStamp>200503081000</hasTimeStamp>
</Context>
    
```

Fig. 3. A segment of OWL code in the CKB

The context knowledge base (CKB) stores the TORA and provides interfaces to context reasoner and context filter. For simplicity, in the following illustration we import the term “triple” from RDF graph [5] to abstract the OWL contextual instances in CKB. E.g. the context in Fig.3 can be abstracted as <Tom locatedIn RoomA (timestamp:200503081000)>.

The context reasoner infers high-level contexts from raw contexts using defined rule set. *The context-aware applications* consume the high-level contexts output from the context reasoner. Each application contains a policy set to choose the reaction to the high-level contexts. The rule set and the policy set can be used as the basis of the attribute-based filtering technology, which will be introduced in Section 6.

The context filter is composed by absolute filter and relative filter. Each application can propose requirements on the absolute attributes of the contexts. If some contexts cannot reach the requirements, they may be discarded by absolute filter without processing. The relative filter analyzes the relevance between applications and contexts to discard the contexts that are not related to applications.

4.2 Main Data Structures

Prior to describing the data structures in ACMR, we assume a scenario of Tom’s mobile phone, which is smart enough to adapt to the user’s contexts. For example, if the user is sleeping, the phone will automatically shuts off the rings; if the user is watching TV, the volume will be turned up.

As mentioned in the previous section, rule set and policy set are two important data structures in the attribute-base filtering technology. The *rule set* is predefined knowledge that is stored in the context reasoner. Its syntax is shown in Fig.4.

```

RuleSet := {Rule}+
Rule := RuleCondition ‘=>’ RuleConclusion
RuleCondition := RuleCondition ^ Triple | Triple
RuleConclusion := Triple
Triple := <subject predicate object>
    
```

(a)

```

type (?u) = Person type(?w) = Washroom
type (?c) = Car
<?u locatedIn ?w> => <?u status UsingWashingroom>
<Person:?u own ?c> ^ <?c status Running> => <?u status driving>
    
```

(b)

Fig. 4. Syntax of rule set

For the sake of simplicity, the disjunction sign “∨” is not available in the syntax because a rule “ A ∨ B => C” can be transformed into two rules: “ A =>C” and “B=>C”. Note that the elements of the triples in rule set are class-based, while the triples in policy engine and the CKB are instance-based. The difference between class-based triples and instance-based triples is that the elements of the former

represent a class while the later only stand for a specific instance. Fig.4b shows the rule set of Tom’s smart phone. In this rule set, ?u stands for all instances of class Person, not a specific instance.

As shown in the Fig.2, each application contains a *policy set*, which help the application to choose the suitable reaction to the environment contexts. The elements of the policy set are pieces of policy. The syntax of the policy is similar with that of the rule in rule set. There are two differences between them: 1.The conclusion (right side of deducing sign “=>”) of policy is not triple but the action to be adopted; 2. The condition (triple in the left side of “=>”) of policy is instance-based. Partial policy set of Tom’s mobile phone is shown in Fig.5. Note that each piece of policy is associated with an absolute attribute requirements list (AARL). This list shows in what situation the corresponding policy is valid. It is reasonable that different policies are associated with different AARL. Take the age attribute for example, some activities are static and some are dynamic. So policies about the dynamic activities require the contexts updated in higher rate. In the following example, watching TV is regarded as the more dynamic activity than sleeping. So the policy about the former limits the contexts should be collected no more than 120 seconds ago, while the limitation of the policy about the later is 600 seconds.

type (Tom) = person	
Policies	AARL
<Tom status WatchingTv> => Volume of Rings Turn Up (1)	Age <= 120 seconds
<Tom status Sleeping> => Rings Shut Off (2)	Age <= 600 seconds Accuracy >= 80%

Fig. 5. Policy set of Tom’s phone

The context filter maintains two important tables for each application: *concerned value (CV) table* and *absolute filtering (AF) table*. Concerned Values, ranging from 0 to 1, map applications to contextual instances. Higher CV denotes the contextual instance is more useful for the application. An example CV table is shown in Tab.1. In the CV table, we define the entry that is both in row R and column C as Entry (R,C).

Table 1. Concerned Value Table in Context filter

APPA	Subject	Predicate	Object
WashroomA			0.05
Tom	0.07		0.03
locatedIN		0.04	

Note that the CV tables record the applications’ concerned value about an instance (the first column of each row) not a context triple. There are 3 columns in each table: subject, predicate and object, which correspond to the three parts of RDF triple respectively. If an instance appears in the different part of a context triple, the application’s CV about it may be different. In above example, the first row means the

AppA's CV about instance WashroomA is 0.05, only if WashroomA is the object of a context. The AppA are not concerned about WashroomA if it is subject or predicate of a context. Similarly, the second row means the AppA's CVs about instance Tom are 0.07 or 0.03, if Tom is subject or object of a context respectively. The application's CVs about unlisted contexts instance are 0. In Section 4 we will presents how can we get and use the CVs.

Table 2. Absolute filtering table

Context	Absolute attribute requirements
<Tom locatedIn Bedroom>	Age <= 600 seconds, Accuracy >= 80%
<TVSet status ON>	Age <= 120 seconds
...	...

AF table records the application's requirements on the absolute attributes of contexts. The format is shown in Tab.2. The left column is the specific context triple and the right one is the application's requirements list of the contexts.

5 Attribute-Base Filtering Technology

The main idea of the Attribute-base filtering follows two principles: (1) If a specific application is concerned about a given context, this context or the high-level contexts deduced from it are likely to exist in the policy set of the application. (2) The applications' requirements on the raw contexts is the same as those on the high-level contexts deduced from them. Based on these two principles, the CV table and AF table can be derived by the policy set and rule set. The attribute-base filtering technology is divided into two processes: 1.creating CV table and AF table; 2.Filtering contexts.

5.1 Creating CV Table and AF Table

Now we will introduce this process step by step:

Step 0: Initialization. The first job in this process is creating a temporary data structure back tracing tree (BTT). The root of the BTT is set as "Policy set" and then the triples in the condition part of policy set are inserted into the BTT as the children nodes of the root. After the initialization of BTT, it is expanded recursively. If some triple (say T) in BTT "matches" a triple (say R) in the conclusion of a rule, the triples in the condition of this rule are inserted into the BTT as the children of T. Note that the triples in the BTT is instance-based, while those in the rule set are class-based. So, here triple T "matches" triple R means T is an instance of R. For example, <Tom status Sleeping> is an instance of <?u status Sleeping>, because the type of "Tom" and "?u" are both Person and "Tom" is a instance of "?u". The final BTT of Tom's phone is shown in Fig. 6.

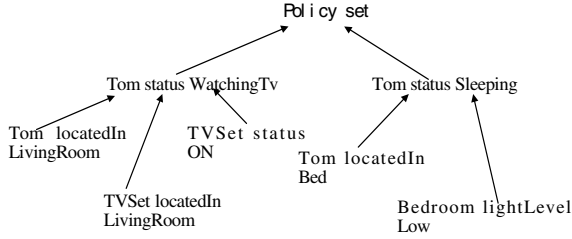


Fig. 6. The final BTT of Tom’s phone

Step 1: Creating CV Table. After creating the BTT, each triple is assigned a CV from top to bottom. The CV of root node is 1. For simplicity, the policies in the policy set and rules in the rule set are assumed to be equally important in this paper. In our future work, we will improve this mechanism by analyzing the importance of a policy. So the CVs of the rest nodes are assigned according to formula 1:

$$CV(M) = CV(Parent(M)) / Num_of_Children(Parent(M)) \tag{1}$$

Fig.7 shows the final CV of each triple node, which is marked beside the corresponding triple.

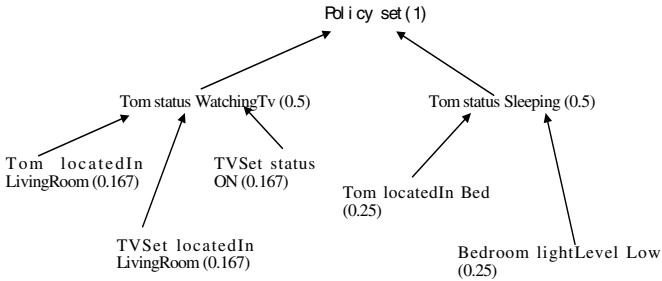


Fig. 7. The BTT marked with CV.

As aforementioned, the triples in BBT are composed of three instances, which are subject, predicate and object respectively. We firstly transform the triples of BBT into Instance-CV triple (instance, part, CV (instance, part)), while the part refers to subject, predicate or object. The CV (instance, part) is assigned as CV (triple) /3. For example, the pair (<Tom locatedIn Bed>, 0.25) is decomposed to three Instance-CV triples: (Tom, subject, 0.083), (locatedIn, predicate, 0.083), and (Bed, object, 0.083). For each Instance-CV triple (I, P, C), the C is added to the Entry (I, P). After above operations, the CV table of Tom’s mobile phone is established. We give partial CV table of Tom’s mobile phone in Tab.2

Step 2: Creating AF table. In this step, we firstly assign the AARL to the children nodes of the root in BTT. Since these triples are the condition part of policy set, they are assigned the AARL associated with the corresponding policies. As the aforementioned principle, the application’s requirements on the raw contexts is the same as those on the high-level contexts deduced from them. Base on this principle, the rest triple nodes inherit the AARL of their parent nodes. By finishing this

Table 3. Partial CV table of Tom's phone.

Tom's phone	Subject	Predicate	Object
Tom	0.472		
locatedIn		0.194	
LivingRoom			0.111
...

assignment, the triple in the BTT is inserted into the left column of AF table and their corresponding AARL is inserted into the right column. If more than one nodes are marked as the same triple and they have different requirements on the same absolute attribute, we choose the most relax one to make sure high successful rate of reasoning. For example, there are two nodes marked "<Tom locatedIn LivingRoom>". If their AARLs declaim "Age <=200 seconds" and "Age <=600 second" respectively, we will choose the later as the final requirement on the "Age" attribute of this triple. The final AF table is shown in Tab.3.

Table 4. Final AF table of Tom's phone

Triple	AARL
<Tom status WatchingTv>	Age <= 120 seconds
<Tom locatedIn LivingRoom>	Age <= 120 seconds
<TVSet locatedIn LivingRoom>	Age <= 120 seconds
<TVSet status ON>	Age <= 120 seconds
<Tom status Sleeping>	Age <= 600 seconds, Accuracy >= 80%
<Tom locatedIn Bed>	Age <= 600 seconds, Accuracy >= 80%
<Bedroom lightLevel Low>	Age <= 600 seconds, Accuracy >= 80%

Step 3: Absolute Filtering. As raw contexts were sent from the CKB to the context filter, the absolute filter matches them with AF table. If one absolute attribute of a context cannot reach the requirement, this context will be discard.

Step 4: Relative Filtering. If a raw context passes absolute filter, its CV is computed by formula (2) firstly.

$$CV \langle S, P, O \rangle = \text{Entry}(S, \text{Subject}) + \text{Entry}(P, \text{Predicate}) + \text{Entry}(O, \text{Object}) \quad (2)$$

If $CV \langle S, P, O \rangle$ is larger than a preset value K , context $\langle S, P, O \rangle$ is regarded as useful information and sent to the context reasoner. Otherwise, it is regarded trashy and discarded. However, the K has to be chosen properly. An overly large K leads to loss of useful contexts, while too small K compromise the efficiency of the context filter. This problem will be discussed in the next section.

Because the context filter discards some contexts, it is possible that the context reasoner cannot infer the right high-level context successfully. In this case, to keep the system available, the context filter is temporarily shut down and all contexts are sent to the context reasoner.

6 Simulation and Discussion

In this section, we will give the results of our prototype experiment. The context reasoner is built based on Jena 2 Semantic Web Toolkit [6], which supports rule-based inference over OWL/RDF graphs. The experiment is run on the Windows 2000 operating system with the P4-2.4GHZ CPU and 256M main memory. To present the ACMR's improvement on the performance, we compare the ACMR with CONON [2] [3], which is an ontology-based context reasoning mechanism without context filter technology. This mechanism is called as CONON because it is used to name the ontology they develop. In our experiments, we continue to use the scenario of Tom's mobile phone and develop 205 OWL classes for it. To demonstrate the ACMR's good performance in case of overwhelming contexts, we simulate a large raw context sets ranging from 500 to 7000 RDF triples in the CKB. The value K in ACMR is set as 0.35 and two systems use the same amount of rules. Fig. 8 shows the comparison between these two systems.

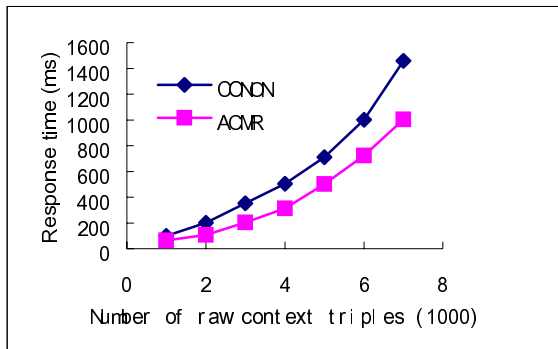


Fig. 8. The comparison of CONON and ACMR

In this figure we can see while the number of raw context triples increase, the advantage of ACMR over CONON becomes larger and larger. It is easily to be explained in theory. Because of the high change rate of contextual information, the bottleneck of the performance in ACMR system is in the context reasoner. As the proportion of useful contextual information is static, the more context triples are sent to the context filter, the more trashy contexts are discarded and the shorter the response time is.

7 Conclusions

In this paper, we present the ACMR, an efficient context modeling and reasoning systems, in which the context ontology TORA is developed to model the contexts and their absolute attributes. Attribute-base context filtering technology is proposed and upgrades the performance of ACMR. In our future work, some improvements are needed in this system, such as analyzing the importance of the rules and policies. However, the experiment shows the good performance of the ACMR system.

References

1. Dey, A.: Providing Architectural Support for Building Context-Aware Applications. PhD thesis, Georgia Institute of Technology (2000).
2. Wang, X.H., Zhang, D.Q., Pung, H.K.: Ontology Based Context Modeling and Reasoning using OWL. Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communication Workshops (2004)
3. Wang, X.H., Dong, J., Chin, C.Y., Hettiarachchi, S.R., Zhang, D.Q.: Semantic Space: An Infrastructure for Smart Spaces. Pervasive Computing, IEEE Vol. 3, Issue 3, July-Sept. (2004)32 – 39
4. Ranganathan, Al-Muhtadi, J., Campbell, R.: Reasoning about Uncertain Contexts in Pervasive Computing Enviroments. Pervasive Computing, IEEE Vol.3, Issue 2, April-June(2004)62 - 70
5. <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>
6. <http://www.hpl.hp.com/semweb/jena2.htm>
7. Lin, X, Li, S. P., Xu, J., Shi, W.: A RDF-based context filtering system in pervasive environment. Proceeding of ICIC 2005 (2005) 306-345
8. Schilit, B. N., Adams, N., Want, N.: Context-aware computing applications. Proceedings of the IEEE Workshop on Mobile Computing System and Application, pages, December , 1994 (1994) 85–90
9. Buchholz, T., Küpper, A. and Schiffers, M. : Quality of Context: What it is and why we need it. Proceedings of the Workshop of the HP OpenView University Association 2003 (2003)
10. K. Henriksen, J. Indulska, and A. Rakotonirainy. Modeling Context Information in Pervasive Computing Systems. Proceedings of the First International Conference on Pervasive Computing (Pervasive 2002), volume 2414 of Lecture Notes in Computer Science (2202) 169–180
11. <http://www.w3.org/TR/owl-ref/>

Influences of Functional Dependencies on Bucket-Based Rewriting Algorithms

Qingyuan Bai^{1,*}, Jun Hong², Hui Wang³, and Michael F. McTear³

¹ Faculty of Mathematics and Computer Science, Fuzhou University, Fuzhou, 350002, China
baiqy@fzu.edu.cn

² School of Computer Science, Queen's University Belfast, Belfast, BT7 1NN, UK
j.hong@qub.ac.uk

³ School of Computing and Mathematics, University of Ulster, Belfast, BT37 0QB, UK
{h.wang, mf.mctear}@ulster.ac.uk

Abstract. There are several algorithms for the problem of query rewriting using views, including the bucket-based algorithms and the inverse rule-based algorithms. The inverse rule-based algorithms have considered the problem in the presence of functional dependencies. However, there are few papers that consider the influences of functional dependencies from the point of view of the bucket-based algorithms. In this paper, we discuss this issue in the bucket-based framework. We mainly consider how to use conditions weaker than the ones in the previous algorithms when forming a set of buckets. The novelty of our approach is that we can enforce additional constraints on the non-exported attributes in a view through functional dependencies. As a result, more valid buckets can be formed. We can then generate additional rewritings which cannot be generated by the previous algorithms. We prove that our algorithm is sound and complete, i.e., the union of the obtained rewriting is a maximally-contained rewriting relative to functional dependencies under certain assumptions. In addition, we also discuss how to construct a recursive rewriting by making use of functional dependencies.

1 Introduction

In a mediator-based data integration system, there are two types of schemas, i.e., a mediated schema and a set of data source schemas. The mediated schema is used to construct queries and describe the contents of data sources. The actual data is however stored in the data sources. Thus, we need to reformulate user queries over the mediated schema into new queries over the data source schemas. Two main approaches are taken to describe the relationships between these two types of schemas, i.e., Global As View (GAV for short) and Local As View (LAV for short). In the GAV approach a mediated schema is defined over data source schemas, while in the LAV approach, data sources are defined over a mediated schema [10]. In this paper we follow the LAV approach because it is easy to delete/add a data source from/to system without making major changes in the mediated schema.

* Supported by the scholarship of Vice-Chancellor of University of Ulster in UK and the Research Foundation of Fuzhou University in China.

The problem of query rewriting using views can be formally given as follows. Given a query Q over a mediated schema, and a set of views $V = \{V_1, \dots, V_n\}$ defined in terms of the mediated schema, a rewriting of Q is a query expression Q' that only uses V_1, \dots, V_n as its operands.

We are interested in finding such a set of rewritings that provide the best possible answers to a query. Many rewriting algorithms have been proposed, e.g., the inverse rule algorithms ([2,4,10]), the SVB algorithm ([8]), and the MiniCon algorithm ([9]). These algorithms are based on the use of either inverse rules or buckets. The inverse rule-based algorithms ([2,3,4]) have considered the problem of query rewriting using views in the presence of functional dependencies (FDs for short). However, there are few papers that consider this problem in the bucket based framework. The following examples show the influences of FDs on query rewriting, i.e., (1) sometimes we need to consider recursive rewritings of a query due to the existence of FDs even though the query is non-recursive, (2) we can enforce additional constraints on non-exported variables through FDs, resulting in more valid buckets to be formed.

Example 1. (from [3]) Suppose we have a relation in a mediated schema `schedule(Airline, Flight_no, Date, Pilot, airCRAFT)`.

Assume that $FDs = \{Pilot \rightarrow Airline; airCRAFT \rightarrow Airline\}$, expressing that pilots work for only one airline, and that any aircraft is used in one airline only.

Hereafter, we use $X \rightarrow Y$ to represent an FD, $X \rightarrow_m Y$ to represent a mapping, and use the rules in Datalog for queries and views ([11]) in this paper. The left hand side of a rule is called the head while the right hand side is the body. Each term in a body is called a subgoal. The variables in the head are distinguished variables and others are existential. An existential variable is shared if it appears in more than one subgoal.

Suppose that the following data source is available:

$V(P, C) :- \text{schedule}(a_1, n_1, d, P, C)$.

Assume a user asks for pilots that work for the same airline as "Mike":

$Q(P) :- \text{schedule}(a_2, n_2, d_2, \text{"Mike"}, c_2), \text{schedule}(a_2, n_2', d_2', P, c_2')$.

The bucket-based rewriting algorithms need to consider a variable mapping from any of the subgoals in Q to V . Note that the variable a_2 is a shared variable in Q and mapped to a_1 , we need to consider a mapping from the subgoals in Q containing a_2 to V . Since V contains one subgoal only, no rewriting can be generated. However, we have a rewriting of Q as follows:

$Q'(P) :- V(\text{"Mike"}, C_1), V(P_2, C_1), V(P_2, C_2), \dots, V(P_n, C_{n-1}), V(P_n, C_n), V(P, C_n), n=1, 2, \dots$

Thus, the first problem that we will address is how to determine in which case a recursive rewriting is needed and then how to generate the recursive rewriting.

Example 2. Suppose that there are three relations $R_1(A_1, A_2)$, $R_2(B_1, B_2)$, and $S(C_1, C_2, C_3, C_4)$ in a mediated schema, and over S two functional dependencies hold, i.e., $FDs = \{S(C_1, C_2, C_3, C_4): C_1 \rightarrow C_2, C_1 \rightarrow C_3\}$. Suppose that we have two views as follows:

$V_1(A, C) :- R_1(A, b_1), R_2(A, b_2), S(C, b_1, b_2, b_3)$.

$V_2(E, F, G) :- S(E, F, G, b_4)$.

Assume that a query is given as follows.

$Q(X) :- R_1(X, y), R_2(X, y)$

Any previous bucket-based algorithms cannot generate any rewritings of Q because:

(1) There is a variable mapping from Q to V_1 , i.e., $\varphi = \{X \rightarrow_m A, y \rightarrow_m b_1, y \rightarrow_m b_2\}$. But it is not guaranteed that $b_1 = b_2$ since both b_1 and b_2 are existential variables in V_1 .

(2) V_2 does not contain any subgoals in Q so that V_2 cannot be used to answer Q .

However, we can get the following rewriting of Q through the given FDs:

$Q'(X) :- V_1(X, C), V_2(C, y, y)$.

Thus, the second problem that we will address is how to enforce additional constraints on existential variables through FDs.

From now on we assume:

1. We only consider the influence of functional dependencies alone.

2. Each view definition should satisfy functional dependencies in a mediated schema, which follows from the papers [2,4,5]. Moreover, we do not consider how to maintain FDs, how to compute the closure of FDs, which are the topics in database systems.

3. We consider only queries without comparisons.

The rest of the paper is organized as follows. In Section 2, preliminaries are given. In Section 3, we have a brief look at related work. In Section 4 we first show how our algorithm works using illustrative examples and then present our algorithm. In section 5, we prove that our algorithm can generate a maximally-contained rewriting relative to FDs under certain assumptions. Finally, we conclude the paper.

2 Preliminaries

A conjunctive query without any arithmetic comparisons has the form:

$$Q(\bar{X}) :- R_1(\bar{X}_1), \dots, R_k(\bar{X}_k) \quad (1)$$

where $R_1(\bar{X}_1), \dots, R_k(\bar{X}_k)$ are called subgoals, also referred to database relations in a

mediated schema. We require that the query be safe, i.e., $\bar{X} \subseteq \bar{X}_1 \cup \dots \cup \bar{X}_k$. A view is a named query. We denote the set of variables in Q and a view V by $\text{Vars}(Q)$ and $\text{Vars}(V)$ respectively.

A query Q_1 is contained in Q_2 , denoted by $Q_1 \subseteq Q_2$, if the answer to Q_1 is a subset of the answer to Q_2 for any database instance. Containment mapping provides a necessary and sufficient condition for testing query containment of conjunctive queries. A mapping φ from $\text{Vars}(Q_2)$ to $\text{Vars}(Q_1)$ is a containment mapping if (1) φ maps every subgoal in the body of Q_2 to a subgoal in the body of Q_1 , and (2) φ maps the head of Q_2 to the head of Q_1 . $Q_1 \subseteq Q_2$ if and only if there is a containment mapping from Q_2 to Q_1 [6,7].

Definition 1 (Contained rewriting). Q' is a contained rewriting of a query Q using the views $V = V_1, \dots, V_n$ with respect to a query language L if the expansion of Q' , denoted by Q'^{EXP} , is contained in Q , i.e., $Q'^{\text{EXP}} \subseteq Q$, where Q'^{EXP} is obtained by

replacing all the views in Q' by their definitions, and existential variables in a view definition are replaced by fresh variables.

Definition 2 (Maximally-contained rewriting [7]). Q' is a maximally-contained rewriting of a query Q using the views V_1, \dots, V_n with respect to a query language L if

- (1) Q' is a contained rewriting of Q , i.e., $Q'^{\text{EXP}} \subseteq Q$,
- (2) there is no other rewriting Q_1 of Q , such that $Q_1^{\text{EXP}} \subseteq Q$ and $Q'^{\text{EXP}} \subset Q_1^{\text{EXP}}$.

An instance of a relation R satisfies the functional dependency $A_1, \dots, A_n \rightarrow B$ if for every two tuples t and u in R with $t.A_i = u.A_i$ for $i=1, \dots, n$, also $t.B = u.B$.

When the relations satisfy a set Σ of functional dependencies, we define our notion of query containment as query containment relative to Σ .

Definition 3 (Query Containment in the Context of FDs).

Query Q_1 is contained in query Q_2 relative to Σ , denoted by $Q_1 \subseteq_{\Sigma} Q_2$, if for each database instance D satisfying FDs in Σ , $Q_1(D) \subseteq Q_2(D)$.

Definition 4 (Maximally-Contained Rewriting in the Context of FDs).

Q' is a maximally-contained rewriting of a query Q using views V_1, \dots, V_n with respect to a query language L relative to Σ , if

- (1) Q' is a contained rewriting of Q relative to Σ , i.e., $Q'^{\text{EXP}} \subseteq_{\Sigma} Q$,
- (2) there is no other rewriting Q_1 of Q , such that $Q_1^{\text{EXP}} \subseteq_{\Sigma} Q$ and $Q'^{\text{EXP}} \subset_{\Sigma} Q_1^{\text{EXP}}$.

3 Related Work

3.1 Algorithms Based on Use of Buckets

A bucket-based rewriting algorithm consists of two steps. In the first step a single-subgoal bucket is created for a subgoal R in a query Q over a view V if the following condition is satisfied (In this case we say V can cover the subgoal R in Q).

(C₁) Any distinguished variables or constants within R in Q should be mapped to distinguished variables or constants in V .

When a shared variable in Q is mapped to an existential variable in V , it needs to form a bucket containing multiple subgoals based on the following condition.

(C₂) A set of subgoals of Q containing a shared variable should be all covered by a single view.

In the second step, rewritings are generated by combining a view from each bucket.

Example 3. We here show the MiniCon algorithm. Suppose that there are three views as follows.

$V_1(A, C) :- r(A, C), s(W_1, C).$

$V_2(B) :- s(B, U_2).$

$V_3(A) :- r(A, W_3), s(W_3, U_3).$

A query is made as follows:

$Q(X) :- r(X, K), s(K, J).$

Using the MiniCon algorithm we form a set of buckets, named MCDs, for each subgoal in Q over $V_i, i=1,2,3$. An MCD is a tuple of the form $(h_C, V(Y)_C, \varphi_C, G_C)$, where h_C is a head homomorphism on V, $V(Y)_C$ is the result of applying h_C to V, φ_C is a mapping from $\text{Vars}(Q)$ to $h_C(\text{Vars}(V))$, and G_C is a set of subgoals of Q. Two types of MCDs are considered, i.e., an MCD containing a single subgoal and an MCD containing multiple subgoals as shown in Table 1.

Table 1. Three MCDs formed in Example 3

h_C	V	φ_C	G_C
$A \rightarrow_m A, C \rightarrow_m C$	$V_1(X, K)$	$X \rightarrow_m A, K \rightarrow_m C$	$r(X, K)$
$B \rightarrow_m B$	$V_2(K)$	$K \rightarrow_m B, J \rightarrow_m U_2$	$s(K, J)$
$A \rightarrow_m A$	$V_3(X)$	$X \rightarrow_m A, K \rightarrow_m W_3, J \rightarrow_m U_3$	$r(X, K), s(K, J)$

After all the MCDs are created, the MiniCon algorithm generates all possible query rewritings by combining views from the MCDs as follows.

Suppose that there are k MCDs, denoted by $C_i, i=1, \dots, k$, such that: (1) $i \neq j, G_i \cap G_j = \emptyset$, and (2) $G_1 \cup G_2 \cup \dots \cup G_k$ = the set of subgoals in Q, where G_i is the field G_C in C_i , then the conjunction of views from $C_i, i=1, \dots, k$, is a contained rewriting of Q. Thus, we can generate two rewritings as follows:

$$Q'(X) :- V_1(X, K), V_2(K).$$

$$Q''(X) :- V_3(X).$$

3.2 Algorithms Based on Use of Inverse Rules

The key idea of the inverse rule-based algorithms [2,10] is to construct a set of rules called inverse rules that invert the view definitions. In the set of inverse rules, the existential variables in the view definitions are replaced with Skolem functions in the heads of the inverse rules. The rewriting of a query is simply the composition of the query and the inverse rules of views using the transformation method [2], the u-join method [10], or the resolution method [4].

In [3], an inverse rule-based algorithm is proposed for query rewriting in the presence of FDs by using a binary relation $e(X, Y)$ to describe an FD. The intended meaning of e is that $e(c_1, c_2)$ holds if and only if c_1 and c_2 must be equal under the given FDs. For example, suppose that there is an FD, $X \rightarrow Y$ over $R(X, Y)$. Then the following rule can be used to represent it.

$$e(Y, Y') :- R(X, Y), R(X', Y'), e(X, X').$$

If a query Q contains the subgoals over which some FDs hold, we need to rectify Q by adding those subgoals that contain only the relation e to Q. After introducing binary relations, a set of chase rules corresponding to a set of FDs are obtained. These chase rules would be used to generate query rewritings associated with a set of inverse rules of views and the rectified query.

A similar approach is proposed in the papers [5]. Grant and Minker [4] also discussed this issue using the resolution method. Suppose that there is an FD, $X \rightarrow Y$ over a subgoal $R(X, Y)$. The following rule is used to represent it:

$$Y = Y' \leftarrow R(X', Y), R(X', Y').$$

This rule is then resolved with a given query and the inverse rules of a set of views to generate query rewritings.

4 Influences of FDs on Bucket-Based Algorithms

As stated in Section 3, the conditions (C_1) and (C_2) are the fundamental criteria of queries containment. We now discuss the influences of FDs on these two conditions.

4.1 Influences of FDs on the Condition (C_1)

The paper [1] first considered the influences of FDs on the condition (C_1) . We here only give a brief review on the algorithm in [1].

Assume that a query Q is given in the form of the formula (1) and a set of FDs is denoted by $\Sigma = \{A_i \rightarrow B_i, A_i, B_i \text{ are attributes in a relation } R_i, 1 \leq i \leq m\}$. The algorithm in [1] first divided all the views into two groups, $V^1 = \{\text{the views containing at least one subgoal in } Q\}$ and $V^2 = \{\text{the views containing no subgoals in } Q\}$. For a view $V \in V^1$, if the condition (C_1) or (C_2) is satisfied, an MCD is formed with $h = \emptyset$ (i.e., null), where h is a partial mapping from $\text{Vars}(Q)$ to $\text{Vars}(V)$. If the condition (C_1) is not satisfied because a distinguished variable (or a constant) in Q is mapped to an existential variable in V , i.e., $h(X) = Z$ (or $h(c) = Z$), where X is a distinguished variable in Q and Z is an existential variable in V , then an MCD is still formed with $h(X) = Z$. Some conditions are given to transform an MCD with $h(X) = Z$ into an MCD with $h = \emptyset$ so that the violation of (C_1) is solved. A rewriting is generated by combining the views in the MCDs with $h = \emptyset$ in the same way as in the MiniCon algorithm.

4.2 Influences of FDs on the Condition (C_2)

We now consider the influences of FDs on the condition (C_2) . The conditions (C_1) and (C_2) are used in isolation. Thus, there are no interactions between the influences of FDs on (C_1) and (C_2) . We first show our approach by an illustrative example and then present our bucket-based algorithm.

Example 4. Continuing with Example 2. When considering the variable mapping from Q to V_1 , i.e., $\varphi = \{X \rightarrow_m A, y \rightarrow_m b_1, y \rightarrow_m b_2\}$, we find that it is not guaranteed that $b_1 = b_2$. However, due to the existence of FDs $= \{S(C_1, C_2, C_3, C_4): C_1 \rightarrow C_2, C_1 \rightarrow C_3\}$, we find the following is a contained rewriting of Q relative to FDs, i.e., $Q^{\text{EXP}} \subseteq_{\Sigma} Q$:

$Q'(X) :- V_1(X, C), V_2(C, y, y).$

In fact, we can get the expansion of Q' as follows:

$Q^{\text{EXP}}(X) :- R_1(X, b_1), R_2(X, b_2), S(C, b_1, b_2, b_3), S(C, y, y, b_4).$

From the FDs, we have: $b_1 = y$ and $b_2 = y$. Hence we have $b_1 = b_2 = y$, which can guarantee that there is a containment mapping from $Q(X)$ to $Q^{\text{EXP}}(X)$.

Thus, the main idea of our approach is that when unifying a set of subgoals of Q with a view V , even though a violation of (C_2) occurs, we still create a bucket **over** V . We then try to find another view U such that the conjunction of U and V can cover the set of subgoals.

For a given query Q in the form of formula (1) and a set of FDs Σ , we use a pseudo-MCD and a general-MCD to store information of unification between Q and a view V .

Definition 5 (A Pseudo-MCD): A pseudo-MCD for a set of subgoals R_1, \dots, R_n of Q over a view V is a tuple of the form (V, h, G_C, CG_C, U) , where:

- V is a view containing the set of subgoals R_1, \dots, R_n .
- G_C is a set of subgoals R_1, \dots, R_n in Q which needs to be all covered by V .
- h is a partial mapping from $\text{Vars}(Q)$ to $\text{Vars}(V)$ that maps a shared variable in R_1, \dots, R_n to different existential variables in V .
- CG_C is a set of subgoals in V which contain other shared variables with G_C .
- U is a view or set of views so that the conjunction of U and V can cover G_C .

Definition 6 (A General-MCD): A pseudo-MCD for a set of subgoals R_1, \dots, R_n of Q over a view V is called a general-MCD if U in a pseudo-MCD has been decided.

A pseudo-MCD containing R_1, \dots, R_n over a view V is formed as follows.

- (1) If (C_1) or (C_2) is true, we build a pseudo-MCD with $h = \emptyset$. In this case, a pseudo-MCD becomes a general-MCD because no violation of (C_1) or (C_2) occurs.
- (2) If the condition (C_2) is not true in the case where a shared variable Y in Q is mapped to different existential variables Z_1, \dots, Z_n in V , we also build a pseudo-MCD with $h = \{Y \rightarrow_m Z_1, \dots, Y \rightarrow_m Z_n\}$. In this case, V cannot cover R_1, \dots, R_n alone. The pseudo-MCD needs to be transformed into a general-MCD. The conditions that ensure a pseudo-MCD to be a general-MCD are given as follows.

Let $CG_C = \{S_1, \dots, S_p\}$ which is a set of subgoals in V containing Z_1, \dots, Z_n as shared variables. If CG_C is empty, we do not need to proceed further.

(p-C₁): $X_1 \rightarrow Z_1, \dots, X_n \rightarrow Z_n$ hold in $\{S_1, \dots, S_p\}$ and X_1, \dots, X_n are distinguished variables in V , and

(p-C₂): There exists such a view or a set of views, denoted by U that contain $\{S_1, \dots, S_p\}$, and $\{X_1, \dots, X_n\}$ and $\{Z_1, \dots, Z_n\}$ are distinguished variables in U .

Then the conjunction of U and V can cover R_1, \dots, R_n , which is proved in Section 5.

4.3 Our Bucket-Based Algorithm

In fact, we can also use the pseudo-MCDs and general-MCDs in Section 4.1 because we can store the violations of (C_1) using a pseudo-MCD and then try to find a view U that satisfies the conditions in [1]. We outline our bucket-based algorithm as follows.

Algorithm BFD: Bucket-Based Query Rewriting in the Presence of FDs

Input: A set of FDs Σ , a set of the views V and a conjunctive query Q .

Output: Q' , a maximally-contained rewritings of Q relative to FDs Σ .

Step 1: FormPseudoMCDs(Q, V)

Form a pseudo-MCD for each subgoal of Q over each view containing at least one subgoal in Q if

(1) the condition (C_1) is satisfied, or

(2) the condition (C_1) is not satisfied because a distinguished variable or a constant

in Q is mapped to an existential variable in a view, or

(3) the condition (C_2) is satisfied, or

(4) the condition (C_2) is not satisfied because a shared variable in Q is mapped to different existential variables in a view.

Step 2: FormGeneralMCDs(C, Σ, \mathbf{MCD})

For each pseudo-MCD whose h is not empty, form a general-MCD by checking the conditions ($p-C_1$) and ($p-C_2$) or the conditions in [1] respectively.

Step 3: CombineGeneralMCDs(\mathbf{MCD})

Generate all the possible query rewritings using the general-MCDs in the same way as the MiniCon algorithm.

End.

4.4 The Need of Recursive Rewritings

As shown in Example 1, due to the presence of FDs, even though a given query is not recursive, its rewritings might be recursive. Now we give the sufficient conditions for the need of recursive rewritings.

CR₁: *There is at least one subgoal R repeatedly appearing in Q, e.g., $R(X_1, \dots, X_k, Z, \dots) \wedge R(X_1', \dots, X_k', Z, \dots)$, and over the subgoal R there are k ($k \geq 2$) FDs whose right hand sides are Zs, i.e., $FDs = \{X_1 \rightarrow Z, \dots, X_k \rightarrow Z\}$.*

CR₂: *There is a view V containing the subgoal R, and X_1, \dots, X_k are distinguished variables, but Z is an existential variable in V.*

If the conditions (CR_1) and (CR_2) are true, then a recursive rewriting of Q is needed when a mapping for the rest of variables satisfies the condition (C_1). Now we construct the recursive rewriting of Q which cannot be generated in the bucket-based framework as follows.

For simplicity, we assume that there are no distinguished variables except $\{X_1, \dots, X_k\}$ in V.

For $i=1, \dots, k$, let $W = V(X_1, \dots, X_i, \dots, X_k) \wedge V(X_1', \dots, X_i, \dots, X_k')$. The expansion of W will contain the following conjunction:

$R(X_1, \dots, X_i, \dots, X_k, Z, \dots) \wedge R(X_1', \dots, X_i, \dots, X_k', Z', \dots)$.

From $X_i \rightarrow Z$, we have $Z=Z'$. Repeating this procedure until all FDs are used, we construct the following conjunction CONJ:

$CONJ = V(X_{1_1}, X_{2_1}, X_{3_1}, X_{4_1}, \dots, X_{k_1}) \wedge V(X_{1_1}, X_{2_2}, X_{3_2}, X_{4_2}, \dots, X_{k_2}) \wedge V(X_{1_2}, X_{2_2}, X_{3_3}, X_{4_3}, \dots, X_{k_3}) \wedge V(X_{1_3}, X_{2_3}, X_{3_3}, X_{4_4}, \dots, X_{k_4}) \wedge \dots \wedge V(X_{1_{k-1}}, X_{2_{k-1}}, X_{3_{k-1}}, X_{4_{k-1}}, \dots, X_{k_{k-1}}) \wedge V(X_{1_k}, X_{2_k}, X_{3_k}, X_{4_k}, \dots, X_{k_k})$.

For $n=1, 2, \dots$, we can construct a recursive query Q_1 as follows:

$Q_1(\dots) = CONJ_1 \wedge \dots \wedge CONJ_n$.

where $CONJ_t, t=1, \dots, n$, is the t-th iteration of CONJ with different subscription. Let ϕ and ψ be the variable mappings from the first R in Q to V and from the second R in Q to V respectively. Then the recursive rewriting of Q, denoted by $Q'(\dots)$, is obtained by replacing $X_1, X_2, X_3, X_4, \dots, X_k$ in the first V in $CONJ_1$ with $\phi(X_1), \phi(X_2),$

$\varphi(X3), \varphi(X4), \dots, \varphi(Xk)$ and replacing $X1, X2, X3, X4, \dots, Xk$ in the last V in CONJ_n with $\psi(X1), \psi(X2), \psi(X3), \psi(X4), \dots, \psi(Xk)$, respectively.

For example, in Example 1, the conditions (CR_1) and (CR_2) are true. We then have:
 $\text{CONJ} = V(P_1, C_1) \wedge V(P_2, C_1) \wedge V(P_2, C_2)$. // $k=2$

We first construct the following recursive query Q_1 :

For $n=1, 2, \dots$

$Q_1 = V(P_1, C_1) \wedge V(P_2, C_1) \wedge V(P_2, C_2) \wedge \dots \wedge V(P_{n-1}, C_{n-1}) \wedge V(P_n, C_{n-1}) \wedge V(P_n, C_n)$.

Because $\varphi(\text{"Mike"})=P_1$ and $\psi(P)=P_n$, we get a recursive rewriting of Q as follows:

$Q'(P) :- V(\text{"Mike"}, C_1), V(P_2, C_1), V(P_2, C_2), V(P_3, C_2), \dots, V(P_n, C_{n-1}), V(P_n, C_n), V(P, C_n)$.

5 Computational Complexity and Correctness of Our Algorithm

5.1 Computational Complexity of the BFD Algorithm

In the first step the BFD algorithm try to form a pseudo-MCD for each view which contains at least one subgoal of Q . Thus, the computation is the same as that in the MiniCon algorithm. In the second step the BFD algorithm needs to check the conditions $(p-C_1)$ and $(p-C_2)$, and the conditions in [1]. In the worst case the total computation in this step is still only polynomial time in size of $\mathcal{O}(k*m*n)$, where k, m, n are the numbers of the subgoals in a query, FDs, and the views respectively. In the final step of the BFD algorithm, the computation is exactly the same as in the MiniCon algorithm. Thus, in the worst case, the total increased computation in our algorithm over the MiniCon algorithm is $\mathcal{O}(k*m*n)$. Our extension of the MiniCon algorithm does not involve a significant increase in computational complexity.

5.2 Correctness of the BFD Algorithm

A maximally-contained rewriting means that if there exist other rewritings, then they are either contained in or equivalent to the maximally-contained rewriting.

Theorem 1. Given a query Q , a set of views V , and a set of FDs Σ in a mediated schema, the BFD algorithm produces a set of contained rewritings of Q , whose union is a maximally-contained rewriting of Q using V relative to Σ .

Proof:

The BFD algorithm solves the following violations:

(Vio1) The condition (C_1) is not satisfied because a distinguished variable or a constant in Q is mapped to an existential variable in a view, or

(Vio2) The condition (C_2) is not satisfied because a shared variable in Q is mapped to different existential variables in a view, or

Note that we do not consider forming a pseudo-MCD over a view V if:

(1) V does not contain any subgoals of Q , or

(2) The condition (C_2) is not satisfied, but $(Vio2)$ does not occur.

The conditions (C_1) and (C_2) are used in isolation so that the influences of FDs on them have not interaction. Now we prove that the BFD algorithm is sound and

complete in such a way that each violation is solved in isolation, i.e., when one of violation is considered, the other does not occur for the same subgoals in Q over the same view.

1. Proof of Soundness

Case 1 (Vio1):

The proof of soundness in this case can be referred to that in [1].

Case 2 (Vio2):

We need to prove that in a general-MCD, there is a containment mapping from the expansion of the conjunction of U and V , $(U \wedge V)^{EXP}$, to Q , i.e., $(U \wedge V)^{EXP} \subseteq_{\Sigma} Q$.

For a pseudo-MCD (V, h, G_C, CG_C, U) , when h is empty, the condition (C_2) is true.

That is, V can cover G_C alone. Now we consider a pseudo-MCD with $h = \{Y \rightarrow_m Z_1, \dots, Y \rightarrow_m Z_n\}$, where Y is a shared variable in R_1, \dots, R_n in Q and Z_1, \dots, Z_n are existential variables in V . Suppose that $CG_C = \{S_1, \dots, S_p\}$ which is a set of subgoals in V containing Z_1, \dots, Z_n as shared variables and $FDs = \{X_1 \rightarrow Z_1, \dots, X_n \rightarrow Z_n\}$ holds in $\{S_1, \dots, S_p\}$. From the conditions $(p-C_1)$ and $(p-C_2)$, we have:

$$V(X_1, \dots, X_n, \dots) :- R_1(\dots), \dots, R_n(\dots), S_1(\dots), \dots, S_p(\dots), \dots$$

$$U(X_1, \dots, X_n, Z_1, \dots, Z_n, \dots) :- S_1(\dots), \dots, S_p(\dots), \dots$$

Then the expansion of $(U \wedge V)^{EXP}$ is:

$$R_1(\dots) \wedge \dots \wedge R_n(\dots) \wedge \underline{S_1(\dots) \wedge \dots \wedge S_p(\dots) \wedge \dots}$$

Due to the existence of $FDs = \{X_1 \rightarrow Z_1, \dots, X_n \rightarrow Z_n\}$ over $\{S_1, \dots, S_p\}$, in the above conjunction with underline we have: $Z_1 = Z_2 = \dots = Z_n$. Thus, the violation of (Vio2) is solved. It means that there is a containment mapping from Q to $(U \wedge V)^{EXP}$.

2. Proof of Completeness

Our proof is similar to the proof of completeness of the SVB algorithm. As mentioned before, a general-MCD is used to solve the violation of the conditions (C_1) and (C_2) . Note that we have two types of the general-MCDs, one has $h = \emptyset$ while another has $h \neq \emptyset$. In the former case V can cover G_C alone while in the latter case $U \wedge V$ can cover G_C . Any view appearing in a general-MCD should be in the field V or the field U in the general-MCD.

We need to show that all the possible rewritings of Q are generated. Let S' be a sound contained rewriting of Q using V that our algorithm did not generate and S' is not contained in any rewriting generated by our algorithm. Let S be the minimized conjunctive query equivalent to S' . There are two cases:

(1) Let us consider the case where there is at least one view $V(X)$ in S that is not in any general-MCDs. Clearly, $V(X)$ is not a redundant subgoal, since S is a minimized version of S' . Each view in a minimized contained rewriting of Q either covers at least one subgoal of Q or the conjunction of the view and other views cover at least one subgoal of Q .

However, since $V(X)$ does not occur in any of the general-MCDs, it must be rejected by the test of forming a pseudo-MCD. That is, $V(X)$ appears in a pseudo-MCD that cannot be transformed into a general-MCD or no pseudo-MCD is formed over V . No matter what the case is, $V(X)$ cannot cover any subgoal of Q alone or

there are no other views so that the conjunction of $V(X)$ and those views can cover at least one subgoal of Q , which is a contradiction to our assumption above.

(2) We now consider the case where a subgoal in S occurs in at least one general-MCD formed by our algorithm. When generating rewritings, we generate all possible combinations of the views from the general-MCDs, which is in the same way as that in the SVB and the MiniCon algorithm. Thus, the proof in the SVB algorithm is still valid. The difference is that there is a conjunction of two or more views to cover a single subgoal of Q , rather than a single view only. However, the conjunction of conjunctive views is still a conjunctive. As a result we conclude a contradiction to our assumption above.

Therefore, if there is any rewriting S of Q that is not generated by our algorithm, it is either not a valid contained rewriting of Q or is contained in at least one of the rewritings generated by our algorithm. \square

6 Conclusions

In this paper, we considered query rewriting using views in the presence of functional dependencies in the bucket-based framework. The novelty of our approach is that we can enforce additional constraints on existential variables through functional dependencies. Specifically, we used conditions weaker than the ones in the MiniCon algorithm when forming buckets. As a result, more valid buckets, named as general-MCDs can be formed. The difference between a general-MCD and an MCD is that in a general-MCD, a set of subgoals in a given query can be all covered by multiple views, while in an MCD the subgoals are covered by a single view only. We can then generate additional rewritings which cannot be generated by the previous algorithms. We prove that our algorithm, the BFD algorithm, is sound and complete, i.e., the union of the obtained rewriting is a maximally-contained rewriting relative to functional dependencies under certain assumptions. We have obtained some experimental results which are omitted here due to the limitation of spaces.

In addition, we discuss how to get a recursive rewriting by making use of functional dependencies. Note that the conditions (CR_1) and (CR_2) are excluded from the BFD algorithm because when they are true, no pseudo-MCDs can be formed. It has been shown in [3] that any non-recursive rewriting with a fixed number of subgoals cannot be maximally-contained rewriting of a query in this case.

References

1. Q. Bai, J. Hong, and M.F. McTear, Scalable Query Reformulation Using Views in the Presence of Functional Dependencies, Proceedings of the 4th International Conference on Advances in Web-Age Information Management, Chengdu, China, Springer-Verlag Press, 2003, pp. 471-482.
2. O. M. Duschka, M. R. Genesereth, Answering Recursive Queries Using Views, Proceedings of the 16th ACM Conference on Principles of Database Systems (PODS'97), Tucson, Arizona, USA, ACM Press, 1997, pp. 109-116.

3. O. M. Duschka, M. R. Genesereth, A. Y. Levy, Recursive Query Plans for Data Integration, *Journal of Logic Programming*, special issue on Logic Based Heterogeneous Information Systems, 43(1): 49-73, 2000.
4. J. Grant, J. Minker, A logic-based approach to data integration, *TLP* 2(3):323-368, 2002.
5. J. Gryz, Query Rewriting Using Views in the Presence of Functional and Inclusion Dependencies, *Information Systems*, 1999, 24(7):597-612.
6. M. Lenzerini, Data Integration: A Theoretical Perspective, *Proceedings of the 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Madison, Wisconsin, USA, ACM Press, 2002, pp. 233-246.
7. A. Y. Levy, Answering Queries Using Views: A Survey, *The VLDB Journal*, 10(4): 270-294, 2001.
8. P. Mitra, An Algorithm for Answering Queries Efficiently Using Views, *Proceedings of the 12th Australasian Database Conference*, Australia, ACM Press, 2001, pp.99-106.
9. R. Pottinger, A.Y. Levy, A Scalable Algorithm for Answering Queries Using Views, *Proceedings of the 26th International Conference on Very Large Data Bases*, Cairo, Egypt, Morgan Kaufmann Publishers, 2000, pp. 484-495.
10. X. Qian, Query folding, *Proceedings of the 12th IEEE International Conference on Data Engineering*, New Orleans, Louisiana, USA, IEEE Computer Society Press, 1996, pp. 48-55.
11. J. D. Ullman, Information Integration Using Logical Views, *Proceedings of the 6th International Conference on Database Theory*, Delphi, Greece, LNCS 1186, Springer-Verlag Press, 1997, pp. 19-40.

GARWM: Towards a Generalized and Adaptive Watermark Scheme for Relational Data

Tian-Lei Hu, Gang Chen, Ke Chen, Jin-Xiang Dong

College of Computer Science, Zhejiang University, Hangzhou 310027, China
andy_hu@263.net, cg@zju.edu.cn, arke@yahoo.com.cn,
djx@zju.edu.cn

Abstract. Watermarking relational data is to protect the intellectual property of sensitive and valuable relational data that is outsourced for sale. Based on analysis of the characteristics of relational data and the corresponding watermarking techniques, the relational data watermarking problem is formalized, and a generalized and adaptive relational data watermarking framework (GARWM) is proposed. The GARWM framework exploits the properties of relational data in the semantics of watermarking, such as preservation of logical relationship in usability preserving attack, discrimination in significance of attributes, and local constraints/global metrics, to strengthen existing methods. The insertion/detection algorithms are presented to insert/detect watermarks into/from non-numeric attributes besides numeric attributes. We identify and classify potential threats to the watermark of relational data including simple attacks, detection-disabling attacks, and ambiguity attacks. We show that GARWM is resilient to these threats, and experiment results quantitatively demonstrate the robustness of our techniques.

1 Introduction

Traditional research on digital watermarking focused on multimedia contents, including image, audio and video[1]. Recently, watermarking text document[2, 3], software[4, 5] and several kinds of discrete data[6] was studied relied upon the combined thoughts and methods from multimedia watermarking and their domain knowledge. Relational scheme, as the de facto fashion of being of data warehouse, data mining, and decision support systems, should also be watermarked to meet the growing needs of IP protection when sensitive, valuable data is outsourced and sold.

There are two major points in the semantics of watermarking digital contents. One is where to hide the watermark, while the other is what contents to protect. By its very nature, relational data differs distinctly from multimedia, text documents, software and semi-structured data in those two points.

Watermarking multimedia data will normally inserted the watermark into high dimensions of the data by transforming it with various methods such as wavelet transformation and FFT; for text documents, the formats (spacing between words and lines) and natural-language semantics of the documents are exploited for embedding watermark; to watermark software, the point is how to watermark the execution procedure/semantics of the software; and watermarking semi-structured data is to protect the data itself, and simultaneously but more pivotally the structure of it.

From relational data's point of view, the location to hide watermark is different from multimedia data and text documents. The watermark can hardly be hidden in the high dimension of data due to lack of redundancy; also it cannot be hidden in the format due to the format of relational data is not as important as in text documents for usability. On the other side, the content to protect in relational data differs from semi-structured data and software. The point to be focused on is to protect the data itself instead of the structure and the usage of data. These characteristics pose unique challenges for relational data watermarking. This paper analyzes these characteristics, and proposes a generalized and adaptive watermark scheme for relational data.

The rest of the paper is organized as follows. Section 2 reviews existing literatures on related topics, and analyzes their limitations and drawbacks. Section 3 formalizes the watermarking problem for relational data, and proposes the GARWM (Generalized and Adaptive Relational data WaterMarking) framework based on analysis of the properties of relational data watermarking. The algorithms for watermark insertion and detection are presented in section 4. Threats to and resilience of the proposed techniques are analyzed in Section 5. Section 6 provides experiments evaluation, and we conclude with summary and future directions in section 7.

2 Related Work

Since Tirkel and et al. clarified the concept of digital watermarking in 1993[7], substantive studies have been conducted on traditional domains on multimedia data (image, video and audio) watermarking[8-11]. The research area is broadened to cover watermarking techniques for text documents, semi-structure data, and software due to growing demands in recent years. For relational data watermarking, two groups simultaneously published their research results. They all concentrate on watermarking numeric relational data and their techniques fundamentally differ from each other.

Agrawal and et al's approach[12] assumes that changes can be made to every attribute of a relation at any least significant ξ bit without significantly reducing the usability. The approach selects a set of tuples in the relation for embedding watermark according to their primary key and a secret key only known by the owner. Generating a pseudo random sequence seeded by those two keys, random bit in random attribute of the tuple is set to random value according to the sequence. We consider this approach as an equivalent to the bit-setter watermarking in multimedia domain.

This scheme is later analyzed by Sion[13], and is considered to be vulnerable to many kinds of usability-preserved attack such as vertical partition and linear transformation in mining association rules. Moreover, this method neglects the usability of watermarked data by a simple but sometime erroneous assumption (the ξ bit assumption above).

More sophisticatedly, Sion and et al's approach[6, 13-15] exploits global constraints to constantly verify the usability, while it utilizes distribution of numeric data for selecting tuples to insert watermark. It chooses only one attribute to embed the watermark to antagonize vertical partition attack. The main algorithm proceeds as follows: The tuples are partitioned into equivalent-sized sets according to a hashed id of the primary key and a secret key, and every set is to insert one bit of the watermark. While embedding a bit into a selected set, the distribution of values in the

set is accounted. Those values in a configured gap are altered to values closed to the standard deviation boundary. The global usability metrics are checked to ensure the usability after watermarking, and if violations occur, the watermarking actions will be rolled back.

This approach can survive from the linear transformation attacks since it makes use of relatively high dimension information, the distribution of data, to insert watermark. However, due to single attribute watermarking fashion, when the watermarked attribute is partitioned out, the whole watermark will be removed. What's more, it relies upon subset markers for embedding watermark, and loss of one subset marker leads to one-bit loss in detecting watermark.

Other literatures on this topic include the watermarking technique based on the cloud model proposed by Zhang and et al.[16] and the NWM scheme proposed by Huang and et al[17]. These proposals can both be classified into the bit-resetting methods similar to Agrawal's approach. Research effort by Gross-Amblard[18] is to investigate the problem of watermarking databases or XML while preserving a set of parametric queries in the usability bound. These literatures are mentioned only for completeness. This paper will mainly based on the former two research efforts, and try to generalize their approach and overcome their shortcomings.

3 The GARWM Framework

The main purpose of GARWM is to provide a comprehensive framework for generalized (not only for numeric data) and adaptive (adaptively watermarks different kinds of data with different methods) relational data watermarking. We will analyze characteristics of relational data concerning watermarking, and propose framework and algorithms for watermark insertion and detection exploiting those characteristics.

3.1 Revisit the Characteristics of Relational Data

A consensus of Agrawal and Sion's work is that the watermark system of relational data should be a blind system. The watermark should be detectable in an outsourced copy of the database relation without knowledge of the original database, since the original database is very likely to be updated since the outsourcing point. However, the blind assumption is not indeed necessary. The data in the relation will be updated with high possibility; whereas the logical relationship, another vital property of relational data, is likely to be invariable to preserve the usability. The none-blindness property can be utilized to counterwork some kinds of attacks, for instance vertical partition, which will be analyzed in section 5.

Attributes to express the relationships, for instance, primary keys and foreign keys, should be omitted when watermarking. We assume that alterations of such attributes will significantly affect the usability, and typically, these attributes do not include sensitive information that is imperative for watermarking (for example the auto-generated integer primary key). Other attributes, which are candidates for watermark embedding, can be classified into several types including at least numeric, text, categorized data, etc. In addition, an attribute has one more property besides its *type* in the semantic of watermark -- the *significance*. Watermark shall be embedded into

attributes with relatively high significance to protect the copyright of sensitive information; however, a tradeoff is that embedding watermark in these attributes will normally lead to significant reduction in usability.

To insert a watermark, target data to be protected have to be modified. A key point for a watermarking technique is to keep the usability while altering the contents. We provide two kinds of metrics to control watermark embedding and evaluate usability of marked data. *Local constraints* are bounds for altering a single attribute, while *global constraints* are to check the usability with a higher-level semantic. Normally, local constraints can be defined as the upper bounds of “the distance” of attributes after/before watermarking, and global constraints will be a series of SQL statements on the watermarked relational data. We introduce the concept of local constraints to reduce possibility of global constraints violations and subsequent rollbacks.

3.2 Formalizations of Relational Data Watermarking

We formally define the problem of watermarking relational data as follows.

Definition 1. (Basic Definitions)

- An attribute is defined as a triple: $A = (t, s, D, f_{lcons})$, where:
 - t is the type of the attribute, s is the significance of the attribute;
 - D is the domain of the attribute; and
 - $f_{lcons} : D \times D \rightarrow \{true, false\}$ is a function to decide whether the watermarked version of data is “closed” enough to the origin version or not.
- S_A is the space consisted by all possible attributes;
- $C = \{m, P, A_1, \dots, A_m\}, m \in \mathbb{Z}, P \in S_A, A_i (1 \leq i \leq m) \in S_A$ is the watermark scheme of the relation, which is a set containing a primary key and a set of m attributes;
- $S_{TUP} = \{\{p, a_1, \dots, a_m\} \mid p \in C.P.D, a_i (1 \leq i \leq m) \in C.A_i.D\}$ denotes the set of all valid tuples in the relation;
- $f_{gcons} : 2^{S_{TUP}} \rightarrow \{true, false\}$ is a function to evaluate the global constraints.
- K is the domain of secret keys used in watermarking; while W is the domain of watermarks to be embedded, which is a distance space;

Definition 2. (The Relational Data Watermarking Problem)

Given $C, f_{gcons}, \mathcal{E} \in \mathbb{R}$, find a watermark function $f_{wm} : 2^{S_{TUP}} \times K \times W \rightarrow 2^{S_{TUP}}$, and a detection function $f'_{wm} : 2^{S_{TUP}} \times K \rightarrow W$, where: $\forall T \subset 2^{S_{TUP}}, k \in K, w \in W$, we calculate the watermarked version $T' = f_{wm}(T, k, w)$, which shall satisfy that:

- **(global cons.)** $|T| = |T'|, f_{gcons}(T') = true$;
- **(local cons.)** $\forall t \in T, \exists t' \in T' \rightarrow t.p = t'.p, \forall 1 \leq i \leq C.m, C.A_i.f_{lcons}(t.a_i, t'.a_i) = true$;
- **(robustness)** $\forall T''$ satisfies $f_{gcons}(T'') = true, f'_{wm}(T'', k) = w', |w, w'| \leq \mathcal{E}$.

3.3 Structure of GARWM Framework

Fig.1. briefly shows the structure of GARWM and the procedures for embedding and extracting watermark.

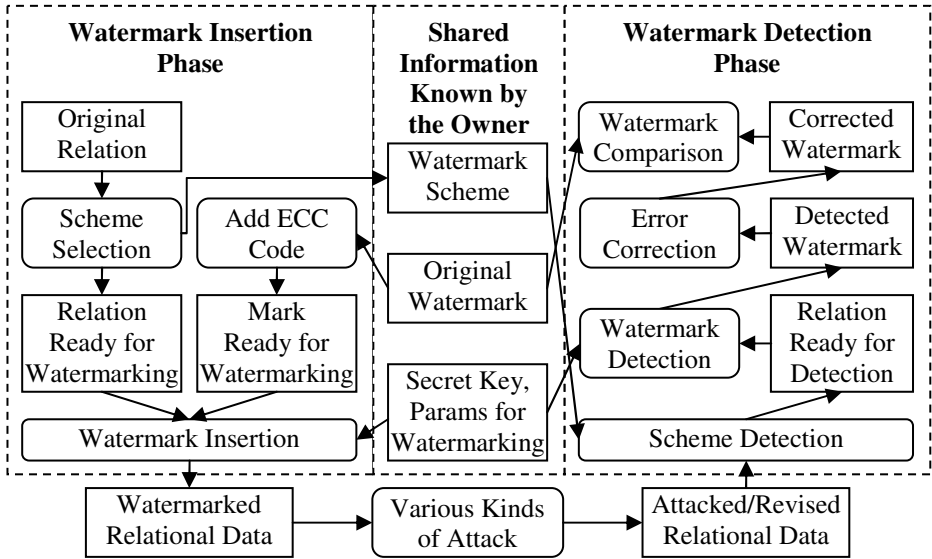


Fig. 1. Structure of GARWM framework. The figure is divided into three parts. The left part is watermark insertion procedures, which take original relation, watermark to embed, and some parameters as inputs, to produce a watermarked version of the relational. The right part is detection procedures, which take a skeptically pirated relation, original watermark, logical scheme of original relation, and the parameters used for insertion as inputs, to judge whether the relation is piratical or not. The central part is shared information that is useful to both insertion and detection phase, and it is possessed by the owner of the relational data.

During the insertion phase, when a piece of relational data is ready for watermark embedding, we (1) select a group of candidates in all attributes of the relation, and recorded it as the watermark scheme (C in Definition 1.); and (2) append the error correction code (ECC Code) to the watermark to embed; and (3) executes the watermark insertion algorithm to watermark the fractional relation with the generated watermark, a provided secret key, and a set of watermark parameters.

During the watermark detection phase, for a piece of relational data which is suspected to be a pirated copy, we (1) extract attributes used in watermarking according to the stored watermark scheme; and (2) execute the watermark detection algorithm to retrieve watermark information from the suspected copy; and (3) exploit error correction mechanism to correct the detected watermark; and (4) compare the corrected version of the watermark with the original watermark to determine whether or not the relational data to be detected is piratical.

Scheme selection and scheme detection exploit none-blindness property mentioned in section 3.1. A partitioned and watermarked attribute shall be found in other relations in the pirated version of relational data, otherwise the usability cannot be preserved. These procedures map every selected attribute in the original relation to an attribute in some relations of the pirated copy. Note that attributes involved in primary key or reference key constraints will be omitted in the candidate selection phase.

ECC code addition and error correction are to recover from minor mistakes in the detected watermark to improve the resilience. We assume that any watermarked data is, with high probability, to be attacked, and an embedded watermark is likely to be altered when attacks take place. Therefore, introduction of error correction mechanism will make the watermark more robust against various kinds of attacks.

Due to limitation of length, we will neglect detailed discussion of these two issues. Instead, the algorithms for watermark insertion and detection, and the resilience analysis will be focused on.

4 Watermark Insertion and Detection

We detail the watermark insertion/deletion algorithms in this section. Before applying the algorithms, the attributes' significance shall be normalized: i.e. to let $\sum_{i=1}^{C.m} C.A_i.s = 1$.

4.1 General Watermark Insertion and Detection

Fig.2 is the algorithm to embed a bit-string watermark wm into a set of tuples T in a relation with watermark scheme C and global metrics function f_{gcons} . key is the secret key only known by the owner.

```

wm_insert ( $C, f_{gcons}, T, key, wm$ )
  initialize  $tup\_groups$  array (size:  $C.m \times wm.length$ ) to all empty buffers;
  foreach  $tup$  in  $T$  : //grouping phase
     $PRS = \text{Random.seed}(tup.p, key)$ ;
     $attr\_idx = \text{attr\_idx\_select}(C, PRS.next() / \text{INTMAX})$ 
     $wmbit\_idx = PRS.next() \bmod wm.length + 1$ 
     $tup\_groups[attr\_idx][wmbit\_idx] \leftarrow tup$ 
  initialize  $wm\_valid$  array (size:  $wm.length$ ) to all false;
  foreach  $i$  in  $[1..C.m]$ ,  $j$  in  $[1..wm.length]$  : // marking phase
     $rollback\_data = \text{wm\_attr\_insert}(C, i, tup\_groups[i][j], wm[j])$ ;
    if ( $f_{gcons}(T)$  equals true) then:  $wm\_valid[j] = \text{true}$ ;
    else: rollback  $rollback\_data$ ;
  return  $wm\_valid$ ;
attr\_idx\_select ( $C, rand$ )
  set  $t = 0.0$ ;
  for  $i=1$  to  $C.m$ :
    if ( $(t += C.A_i.s) \geq rand$ ) then: return  $i$ ;
  return  $C.m$ ;

```

Fig. 2. Watermark insertion algorithm

In the first part (grouping phase), for each tuple in the relation, the algorithm first seeds a pseudo random sequence by the primary key and the secret key. We use the sequence to determine which attribute shall be marked based on the significance of the attributes in the relation. Thus, the algorithm will more probably select an attribute with relatively higher significance for watermark insertion. Next, the bit of watermark

to be inserted is selected based on the random sequence. Record the tuple in the tuple buffer associated with the selected attribute and bit index.

In the next part (marking phase), each tuple buffer is processed by the attached watermark function in turn. After the tuple buffer is processed, the global metrics is accounted. In case the metrics are violated, the alterations are rollbacked. The valid map to indicate which bits are successfully embedded is returned. It will be used in watermark comparison after detection procedure.

The detection algorithm first groups the tuples in T' the same as the grouping phase in insertion algorithm. Each generated tuple group determines a bit in the detected watermark by the attached detection function. In case there are two or more groups determine the same bit in the watermark by different attribute, the majority voting mechanism is used for deciding the final watermark bit. The watermark is constructed after all tuple groups being processed. We omit detailed procedure here.

4.2 Particular Type Processing

In order to support various attribute types in watermarking, we proposed a mechanism to bidirectionally map the attribute with a single numeric value. For the i th attribute $C.A_i$, we define two functions: $f_{\rightarrow}: C.A_i.D \rightarrow \mathbb{R}$ and $f_{\leftarrow}: \mathbb{R} \rightarrow C.A_i.D$ for the bidirectional map. Thus, watermarking data in various types is reduced to watermarking numeric data.

Two most common types besides numeric type in relational data are concerned: categorical data and text data. Categorical values have a countable domain, and we define f_{\rightarrow} and f_{\leftarrow} by marking each possible value in the domain with an integer index.

Distinct from numeric data and categorical data, the usability metrics for text data should be the human-readability instead of the machine-readability. Thus, we identify the bandwidth to embed watermark into relational text data to be the synonyms and the homographs. For example, in database semantics, “Table” is a possible substitute for “Relation”; however, “Relation” and “Relation” are also candidates for substitution. Thus, f_{\rightarrow} and f_{\leftarrow} can be generated based on a synonyms dictionary and a homographs-mapping table (e.g. maps ‘l’ with ‘l’ and ‘l’, maps ‘O’ with ‘O’, etc.). Note that, for alphabetical language, the homographs can be differentiated by simple methods (e.g. converting all characters to uppercase/lowercase), and techniques exploiting them do not satisfy the “imperceptibility” property required by digital watermark. However, for ideographic languages, such as Chinese, the homographs can actually be utilized to embed watermark due to the difficulty in differentiation.

Fig.3 is the algorithm to insert a bit (wm_bit) of watermark into a group of tuples ($tupbuf$) at specified attribute (i) by a differential criteria σ controlled by the owner. For numeric number set N , denote: $cnt(N)$ and $avg(N)$ are individually the item count/average value of N , $GT(N)$ and $LT(N)$ are individually subsets of items whose values are greater/less than $avg(N)$. When bit ‘1’ being inserted, the numbers in N are altered to make $cnt(GT(N)) - cnt(LT(N)) \geq \sigma * cnt(N)$; otherwise, the algorithm makes $cnt(LT(N)) - cnt(GT(N)) \geq \sigma * cnt(N)$. The per-attribute detection can be derived from the insertion procedure, and the criteria to judge whether ‘1’ or ‘0’ is embedded is whether $cnt(GT(N)) \geq cnt(LT(N))$ or not; we omit details for concision.

```

wm_attr_insert ( $C, i, tupbuf, wm\_bit$ )
  empty set  $N$ , and  $rollback\_data; A = C.A_i;$ 
  foreach  $tup$  in  $tupbuf$ :
     $N \leftarrow val = A.f_{\rightarrow}(tup.a_i);$ 
    associate  $\&val$  with the corresponding tuples by function  $tm$ 
  if ( $wm\_bit$  equals 1) then: // alter values towards the more-than-avg direction
    while ( $cnt(GT(N)) - cnt(LT(N)) < \sigma * cnt(N)$ ):
       $curr\_buf \leftarrow$  select  $\sigma * cnt(N)$  items from  $LT(N)$ ;
      foreach  $val$  in  $curr\_buf$ :
         $newval \leftarrow$  alter  $val$  to be larger than  $avg(N)$ ;
        if ( $A.f_{icons}(A.f_{\leftarrow}(val), A.f_{\leftarrow}(newval))$  equals true)
          then:  $*(&val) = newval$ .
    else: almost the same except that altering towards the less-than-avg direction
  foreach altered  $val$  in  $N$ :
     $rollback\_data \leftarrow$  alter_tuple ( $tm(\&val), i, f_{\leftarrow}(val)$ );
  return  $rollback\_data$ ;

```

Fig. 3. Per-attribute watermark insertion algorithm

5 Resilience Analysis

Resilience, i.e. robustness, is the vital property for watermarking techniques. In this section, we will first categorize the possible threats to relational data watermark, and then analyze the resilience of GARWM to those threats.

5.1 Taxonomy of Threats to Relational Data Watermark

Take popular appellations from literatures on cryptography that: Alice, the owner of relational data, has watermarked it before distribution; Mallory is the adversary who wants to pirate the IP of Alice. In conformity to the attack classification in multimedia watermarking literatures[1, 19], we classify the threats to relational data watermark as following:

- **Simple attacks:** Mallory transforms tuples in original relation T distributed by Alice to impair the watermark. There are two kinds of attacks in this category:
 - *Uniform transformation.* Mallory produces T' by uniformly transform all tuples in T . A normal case is linearly transformation on numeric attributes.
 - *Random modification.* Mallory produces T' by randomly modify some tuples in T . Normal cases are random space padding on text attributes.
- **Detection-disabling attacks:** Mallory breaks correlations between original relation T and embedded watermark, and makes the recovery of watermark infeasible.
 - *Tuple insertion.* Mallory inserts a series of new tuples into T .
 - *Tuple deletion.* Mallory deletes a series of insignificant tuples from T .
 - *Horizontal partition.* Mallory reserves a series of important tuples in T , and horizontally partitions other tuples into other relations.
 - *Multi-relation merge.* Mallory joins two or more relations including T on some attributes, normally the primary key.

- *Vertical partition.* Mallory vertically partitions some unimportant attributes of relation T to other relations, or just removes those attributes.
- **Ambiguity attacks:** Mallory tries to insert/identify his watermark into/from T , in order to claim the ownership over the relation falsely.
 - *Watermark addition.* Mallory embeds additional watermark(s) into T .
 - *Spurious watermark identification.* Mallory identifies a random but fake watermark in T , and claims that he inserts this watermark.

Note that the “removal attacks” category identified in [1] is omitted, since those attacks in this category either cannot be applied to relational data watermark, or can be classified into other categories.

5.2 Resilience of GARWM

For *uniform transformation* attacks, a most common case is, for numeric data, the linear transformation attack, which maintains the usability in mining association rules [13]. Such attacks do not inflect the distribution, so $cnt(GT(N)) - cnt(LT(N))$ is fixed applying such transformation, i.e. the detected watermark will not be altered.

For *random modification* attacks: randomly bit-setting/rounding attack on numeric data always modifies the least significant bits (LSB) of a numeric number. These minor modifications do not affect the macroscopical distribution of the whole data set, and the value of $cnt(GT(N)) - cnt(LT(N))$ will maintain fixed or vary in an acceptable bound. The randomly blank/invisible characters padding/inserting attack on text data do not affect the value (after applying the f_{\rightarrow} transformation) used to embed watermark, which leads to no effect on the detection of embedded watermark.

Other random modification attacks, including random synonymous/homomorphous words substitution for text attributes and semantics-preserving alteration for categorical attributes, modify not only the LSB but also MSB when transforming to numeric values. In addition, the tuple insertion, tuple deletion, and horizontal partition attacks, categorized into the *detection-disabling* threats, do not affect the attributes in existing tuples; instead count of values in calculating the distribution will be affected. These kinds of attacks will probably modify or remove the embedded watermark; however, we will show the probability of modifying the watermark is extremely low.

Assume that the error correction mechanism can correct $1/\tau \times |wml|$ bits of watermark, and the probability of one-bit loss after modifying/inserting/deleting a percentage of φ tuples is $P_1(\varphi, \sigma)$ before error correction, where σ is the differential criteria introduced in section 4.3. According to Bernoulli Trials, the probability of actually one-bit loss after error correction is:

$$P(\varphi, \sigma) = 1 - \sum_{i=0}^{1/\tau \times |wml|} C_{|wml|}^i \times P_1(\varphi, \sigma)^i \times (1 - P_1(\varphi, \sigma))^{|wml|-i} \tag{1}$$

Since $P_1(\varphi, \sigma)$ is closely related to the distribution of data and the alteration on data, we will give a quantitative analysis based on the experiment results in section 6. For demonstration purpose here, taking the results from section 6, when σ equals to 8‰ and 3/4 tuples in the watermarked relation is removed/inserted/modified, $P_1(\varphi, \sigma)$ is respectively 1%/2%/5%, and given $\tau=10$ for a 20-bits sized watermark, the possibility of actually one-bit watermark loss is: 0.1%/0.7%/7.5%.

Our solution naturally survives the vertical partition and multi-relation mergence attack: the candidate selection and candidate scheme detection mechanism stated in section 3.3 will ensure the one-one map between the attributes in T and T' .

The confront remedies for the ambiguity attacks are not specified to the relational data, refer to [1] for details about the those techniques exploiting trusty third-party timestamps and noninvertible watermarks.

6 Experiments

The experiments are conducted on a relation with 200000 tuples, and for the evaluation purpose, the relation is created containing only one numeric attributes besides the primary key. The values in the attribute are evenly distributed, and the watermark to be embedded is 20-bits in length. No error-correction mechanism is provided for simplicity, and the usability constraints are ignored. Each result shown is an average of 100 times running of each experiment.

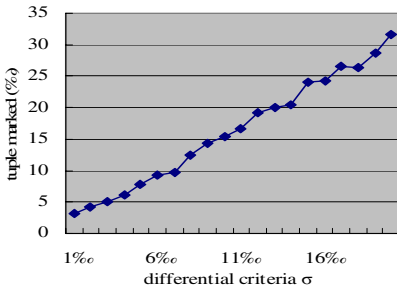


Fig. 4. Tuple alteration ratio vs σ

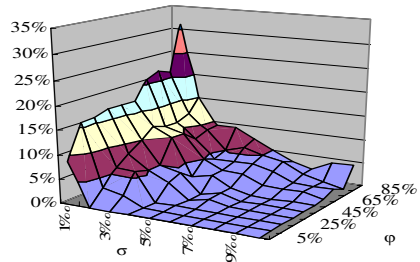


Fig. 5. Random data loss resilience

Fig.4 shows the relationship between the tuple alteration ratio and the differential criteria σ . While increasing the differential criteria, the percentage of marked tuples increases. Normally the more tuples are marked, the less usability of the relation is.

To evaluate the resilience of GARWM in a quantitative way, we conduct experiments to simulate three kinds of attacks, includes:

- Random data loss: randomly remove a percentage of ϕ tuples.
- Random data addition: randomly insert a percentage of ϕ tuples.
- Random data alteration: randomly modify a percentage of ϕ tuples.

Fig.5 shows the possibility (i.e. the $P_1(\phi, \sigma)$) of one-bit mark loss when random data loss occurs. The X-, Y-, and Z- axis are individually the differential criteria σ , the tuple removal ratio ϕ , and the possibility. We evaluate random data loss from very few (5%) to the majority (95%) for σ from 1‰ to 10‰. While σ increases, the possibility drops down sharply, and when σ nears 8‰, the possibility of one-bit watermark loss is very low to only 1% when 75% tuples are removed. For a certain σ , more data loss leads to higher possibility of watermark loss; however, compared to the increasing rate of data loss, the increasing rate of loss possibility is relatively low.

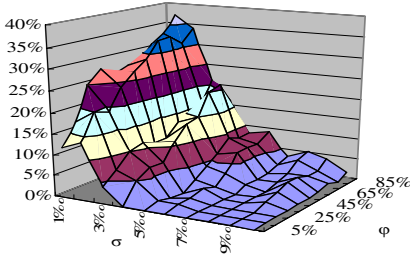


Fig. 6. Random data addition resilience

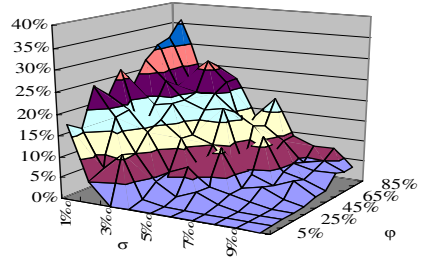


Fig. 7. Random data alteration resilience

Fig.6 and Fig.7 are the possibilities of one-bit mark loss individually for random data addition and random data alteration. Conclusions can be drawn almost the same as the random data loss attack, since when σ nears 8%, the possibility of one-bit watermark loss is only 2%/5% individually when 75% tuples are inserted and modified.

These results provide solid validation to the robustness of our techniques.

7 Conclusion

In this paper, the problem of relational data watermarking is introduced. We

1. proposed GARWM framework, a generalized and adaptive solution for relational data watermarking, which applies to various types of data besides numeric data, and
2. presented algorithms for watermark insertion and detection exploiting the distribution of numeric values and map functions between data in non-numeric types and numeric type, and
3. categorized various kinds of threats to relational data watermark, and
4. analyzed the robustness of GARWM to those threats qualitatively and quantitatively.

Comparing with existing proposals, GARWM is robust to linear numeric transformation and vertical partition attacks compared to Agrawal’s approach. It does not rely on sequence of primary key and subset markers, which makes the approach more resilient to random data loss/addition attacks compared to Sion’s approach.

Further work includes identifying other possible threats to relational data watermark, modeling the attacks formally, and perfecting the framework and the underlying algorithms.

References

1. Hartung, F. and Kutter, M., *Multimedia Watermarking Techniques*. Proc. of the IEEE (USA), 1999. **87**(7): p. 1079-1107.
2. Chiang, Y.L., Chang, L.P., et al., *Natural Language Watermarking Using Semantic Substitution for Chinese Text*. In: Kalker, T., Cox, I.J., and Ro, Y.M. (eds). *Int. Workshop on Digital Watermarking, IWDW*. 2003. Seoul, Korea: Springer. p. 129-140.

3. Kim, Y.W., Moon, K.A., and Oh, I.S., *A Text Watermarking Algorithm Based on Word Classification and Inter-Word Space Statistics*. In: (eds). *Int. Conf. on Document Analysis and Recognition*. 2003. Edinburgh, Scotland. p. 775-779.
4. Collberg, C., Debray, S., et al., *Dynamic Path-Based Software Watermarking*. In: (eds). *ACM SIGPLAN Conf. on Programming Language Design and Implementation*. 2004. Washington DC, USA: ACM Press. p. 107 - 118.
5. Nagra, J., Thomborson, C.D., and Collberg, C.S., *A Functional Taxonomy for Software Watermarking*. In: Oudshoorn, M.J. (eds). *Australasian Computer Science Conf., ACSC*. 2002. Monash University, Melbourne, Victoria: Australian Computer Society. p. 177-186.
6. Sion, R., *Rights Assessment for Discrete Digital Data*. Ph.D. Thesis, Purdue University, 2004.
7. Tirkel, A.Z., Rankin, G.A., et al., *Electronic Water Mark*. In: (eds). *Digital Image Computing, Technology and Applications, DICTA*. 1993. Macquarie University, Sidney. p. 666-673.
8. Swanson, D., M., et al., *Transparent Robust Image Watermarking*. In: (eds). *SPIE Conf. on Visual Communications and Image Proc*. 1996. p. 211-214.
9. Boney, L., Tewfik, A.H., and Hamdy, K.N., *Digital Watermarks for Audio Signals*. In: (eds). *Int. Conf. on Multimedia Computing and Systems*. 1996. p. 473-480.
10. Cox, I., Kilian, J., et al., *Secure Spread Spectrum Watermarking for Multimedia*. IEEE Trans. on Image Processing, 1997. **6**(12): p. 1673-1687.
11. Hartung, F. and Girod, B., *Watermarking of Uncompressed and Compressed Video*. Signal Processing, 1998. **66**(3): p. 283-301.
12. Agrawal, R., Haas, P.J., and Kiernan, J., *Watermarking Relational Data: Framework, Algorithms and Analysis*. The VLDB Journal, 2003. **12**(2): p. 157-169.
13. Sion, R., Atallah, M., and Prabhakar, S., *Rights Protection for Relational Data*. IEEE Trans. on Knowledge and Data Engineer, 2004. **16**(6): p. 1-17.
14. Sion, R. and Atallah, M., *Attacking Digital Watermarks*. In: (eds). *SPIE Symposium on Electronic Imaging*. 2004. San Jose. p. 848-858.
15. Sion, R., Atallah, M.J., and Prabhakar, S., *Power: A Metric for Evaluating Watermarking Algorithms*. In: (eds). *Int. Conf. on Information Technology: Coding and Computing, ITCC*. 2002. Las Vegas, Nevada. p. 95-99.
16. Zhang, Y., Zhao, D.N., and Li, D.Y., *Watermarking Relational Databases (in Chinese with English Abs.)*. Journal of PLA University of Science and Technology, 2003. **4**(3): p. 1-5.
17. Huang, M., Cao, J., et al., *A New Watermark Mechanism for Relational Data*. In: (eds). *Int. Conf. on Computer and Information Technology, CIT*. 2004. Wuhan, China. p. 946-950.
18. Gross-Amblard, D., *Query-Preserving Watermarking of Relational Databases and Xml Documents*. In: (eds). *ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. 2003. San Diego, California. p. 191-201.
19. Hartung, F., Su, J.K., and Girod, B., *Spread Spectrum Watermarking: Malicious Attacks and Counterattacks*. In: (eds). *SPIE Security and Watermarking of Multimedia Contents*. 1999. San Jose, CA. p.

Refined Rules Termination Analysis Through Activation Path

Zhongmin Xiong¹ and Zhongxiao Hao^{1,2}

¹ School of Computer and Control, Harbin University of Science and Technology, Harbin 150080, P.R. China

² Department of Computer and Engineering, Harbin Institute of Technology, Harbin 150001, P.R. China

zhmxiong123@yahoo.com.cn, zhmxiong@sina.com

Abstract. Supporting active rules has become an important characteristic of modern database systems. While active rules termination is an undecidable problem, several recent works have suggested proving termination by using triggering and activation graphs. In this paper, we propose such concepts as activation reachable, activation path, inhibited activation cycle, inhibited activation rule, effective activation path to refine these works. Based on these concepts and the formula constructed for an effective activation path, we show how to determine the termination of active rules set. The analytical results prove that our approach can detect more termination situations than existing works. Moreover, the proposed approach is practical and general with respect to the various rule languages from Event-Condition-Action language for XML to the trigger language in SQL: 1999 standard. As a result, it can be easily applied to modern database systems.

1 Introduction

Active rules are used to provide reactive functionality in many areas, including active databases [1,2,3], workflow management [4], and XML documents [5,6].

Active rules are structured according to paradigm Event-Condition-Action. As the way to specify a rule set is unstructured, the termination problem (the rules execution can sometimes be infinite) of a rule set is known as an undecidable problem [7].

Many works propose some sufficient conditions allowing to guarantee termination. Some researches [1,8] propose their analysis based on *triggering graph* (TG), while others [2,3,9-14] refine TG analysis considering the condition part of the rules. [2,3] augments a TG with an *activation graph* (AG) where an edge will be removed unless it is in a cycle or reachable from a cycle and it can be re-activated after a self-deactivation. [9] construct a conjunct based on the trigger conditions of two rules. If the conjunct is not satisfiable, the edge between the two rules will be removed. [10,11] construct a stronger condition for termination decision than [9]. [12] proposes a sophisticated technique for building Activation Graphs, which complements nicely with the techniques presented in [2,3]. [13] translates ECA rules into deductive rules. However, proving termination requires the equivalence of translation from ECA rules to deductive rules. [14] introduces a method based on Abstract Interpretation, which

uses large complex tables to examine all properties. [5,6] study ECA rules in the context of XML data. [5] does not focus on analysis of the rules' behavior, while [6] describes an ECA rule language for XML and offers the techniques for inferring triggering and activation relationship between pairs of its XML ECA rules.

In this paper, we exploit TG and AG to solve the rules termination problem. With respect to various rule languages, TG and AG can be constructed by methods from [6,12]. The proposed approach in this paper is both practical and general because it is independent of the various rule languages. Moreover, it improves the techniques introduced by [2,3,9-11] and is more efficient and effective than methods in [13,14]. In this paper, such notions as activation reachable, activation path, inhibited AG cycle and inhibited activation rule are introduced. Under our assumption of active rules' execution semantics presented in section2, we observe two cases as follows.

Case1: For a path in TG or in AG, if its formula constructed by methods in [9-11] is not satisfiable, such a path must not be activable. [9-11] only partly consider this case because they do not take into account the activation relationship between rules, while [2,3] do not consider this case at all.

Case2: For any rule in a TG cycle, if no activation path can be synchronously and infinitely executed with it, it must be terminable. [2,3,9-11] do not consider this case.

The remainder of this paper is structured as follows. Section2 introduces the preliminaries for our work; Section3 presents three motivating examples; Section4 proposes such notions as activation path and activation path set; In section5, we show how to analyze the termination of a TG cycle using our methods; Section6 provides our algorithm for termination analysis; Section7 concludes the paper.

2 Preliminaries

2.1 Active Rules

The active rules are structured according to paradigm Event-Condition-Action. *Event* is either data operation event (inside of the Database system) or event reported to the system by the outside. *Condition* is a request on the database and is always expressed as database queries. *Action* is generally composed of a sequence of database updates, or a procedure containing database updates.

Let r_i and r_j be two rules, we consider two different relations among rules.

- r_i triggers r_j , if one of the r_i 's actions has the corresponding event in r_j ;
- r_i activates r_j , if the possible execution of the r_i 's actions makes r_j 's condition true;

These relations can be described with *Triggering Graph* (TG) and *Activation Graph* (AG), respectively. TG can be built by means of a syntactic analysis of rules [1], and AG can be built by the techniques provided by [12].

In this paper, all rules in a rule set should be self-disactivating rule, whose action's execution falsifies its condition. Thus, the AG can provide additional information for termination analysis with respect to the TG [2,3].

[6] describes an ECA rule language for XML and offers the techniques of building TG and AG for these XML ECA rules.

2.2 The Rule Execution Model

Coupling modes allow to specify the evaluation moment of the Condition Part (E-C coupling), or the execution moment of the Action Part (C-A coupling). The most frequent modes are *Immediate* and *Deferred*. In this paper, E-C coupling and C-A coupling use *immediate* mode, in which case the condition (action) is evaluated (executed) immediately after the event (condition).

Cycle policy addresses the question of what happens when events are signaled by the evaluation of the condition or action of a rule. Database systems always support a *recursive cycle policy* for *immediate* rules. In this case, event signaled during condition and action evaluation cause the condition or action to be suspended. Thus, any *immediate* rules monitoring the events can be processed at the earliest opportunity.

The *scheduling* phase of rule evaluation determines what happens when multiple rules are triggered at the same time. In this paper, *scheduling* is *All Sequential*. Thus, the system fires all rule instantiations sequentially.

In this paper, the *rules process* is described as follows. When a rule is triggered, the system evaluates its condition. If the condition is satisfied, the rule is removed from the triggered rules set and its action is performed. If the condition is not satisfied, then *event-consuming* policy is adopted. That is, the system eliminates the rule from the triggered rules set.

Details about the Rule Execution Model presented here can be seen in [15,16].

3 Motivating Examples

According to Oracle8.1.7 DDL [17], all active rules in these examples are described as Oracle’s triggers which are self-disactivating rules [2].

Example 1. Figure1 presents the triggering and activation graph for a rule set, where the solid arcs are triggering ones and the dashed arcs are activation ones. Since each

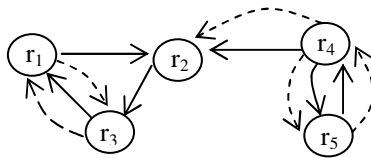


Fig. 1. Triggering and Activation Graph for Example1

rule is in a TG cycle and can be re-activated after a self-deactivation, [2,3] determine this rule set may be non-termination. $\{r_4, r_5\}$ is the only cycle in both TG and AG. According to [2,3], it is concluded that r_1 and r_3 should be reachable from $\{r_4, r_5\}$. But this conclusion cannot be obtained based on the notion of a rule is reachable from a rule set [2,3]. This is because there are no two rules r_a and r_t that both can be reachable from $\{r_4, r_5\}$, making $\langle r_a, r_t \rangle \in AG$ and $\langle r_t, r_j \rangle \in TG$ where r_j is r_1 or r_3 .

Example 2. Six triggers are defined on four tables: R (A, B, C, D), S (H, L), Q (I, F), T (E, G, M) and they are shown in Figure2.

```

create trigger r1          create trigger r2          create trigger r3
after update of A on R   after update of H on S   after update of C on R
when R.B=1              when (S.H=0) and (T.M=1) when R.D=1
begin                  begin                  begin
  update table S         update table R set C=1;  update table R
  set H=0,L=1;          update table S set H=1;  set A=0,B=1,D=0;
  update table R         update table T          end;
  set B=0;              set E=1,G=0;
end;                   end;

create trigger r4        create trigger r5        create trigger r6
after update of G on T   after update of I on Q   after update of F on Q
when (T.E=1) and (T.M=0) when Q.I=0              when Q.F=1
begin                  begin                  begin
  update table Q set I=0; update table Q          update table Q
  update table T set E=0; set I=1,F=1;          set F=0;
  update table R set D=1; end;                  end;
end;                   end;
    
```

Fig. 2. Example2 with Oracle’s triggers.

Figure 3 illustrates a TG cycle $R_1 \{r_1, r_2, r_3\}$ and an AG cycle $R'_1 \{r_1, r_2, r_4, r_3\}$. As each rule is in R_1 or can be reachable from R_1 and it can be re-activated after a self-deactivation, [2,3] determine that this rule set may be non-termination. $T.M$ is the only Non-updatable attribute and no finitely-updatable attribute can be detected by [11]. The path $\langle r_2, r_4 \rangle$ is not activable, since the formula along this path can be constructed by [9]: $(T.M=1) \text{ AND } (T.M=0)$ and it is not satisfied. Only one path can be removed from TG by [9-11], and the termination cannot be guaranteed. However, R'_1 is a “false” AG cycle due to non-activation of $\langle r_2, r_4 \rangle$. Therefore, no rule can be guaranteed to be re-activated after a self-deactivation. As a result, this rule set always terminates. No previous algorithm deals with the activation formula of a path in TG and a path in AG at the same time, so they cannot detect such a termination case.

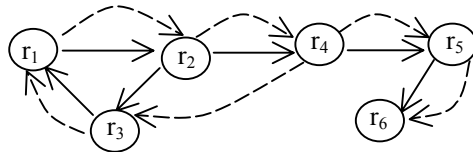


Fig. 3. Triggering and Activation Graph for Example2

Example 3. Six triggers are defined on four tables: R (A, B, C, D), S (H, L), Q (I, F), T (E, G). r_1, r_3 and r_5 take the same definitions as in Figure2, but r_2, r_4 and r_6 are defined as Figure 4. Figure 5 illustrates two TG cycles: $R_1 \{r_1, r_2, r_3\}, R_2 \{r_4, r_5, r_6\}$ and an AG cycle $R'_1 \{r_1, r_2, r_4, r_3\}$.

<pre> create trigger r2 after update of H on S when S.H=0 begin update table R set C=1; update table S set H=1; update table T set E=1; end; </pre>	<pre> create trigger r4 after update of G on T when T.E=1 begin update table Q set I=0; update table T set E=0; update table R set D=1; end; </pre>	<pre> create trigger r6 after update of F on Q when Q.F=1 begin update table Q set F=0; update table T set G=0; end; </pre>
---	---	---

Fig. 4. Example3 with Oracle’s triggers

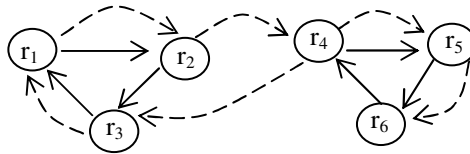


Fig. 5. Triggering and Activation Graph for Example3

Under our assumption of rule execution semantics as above, R_1 and R_2 cannot be synchronously executed at the same time. When any rule in R_1 is triggered by the system, r_4 cannot be triggered at the same time. After a self-deactivation, the condition of r_3 is false without the re-activation by r_4 . R_1 is termination due to the termination of r_3 . This rule set is termination at this time due to no non-termination TG cycle. The same analytical result can be obtained when any rule in R_2 is triggered by the system. Thus, this rule set always terminates.

All attributes in the conditions of this rule set are updatable and no finitely-updatable attribute can be detected by [11]; No edge can be removed from TG by [2,3]. So [2,3,9-11] cannot deal with such a termination case.

4 Activation Path and Activation Path Set

From Example1, we can see the inadequacy of the notion in [2,3]. To improve them, we present Definition 1 and Definition 2.

Definition 1. A rule r_j is *directly triggering reachable* from a rule set R if there is a rule r_i in R such that $\langle r_i, r_j \rangle \in TG$; a rule r_j is *triggering reachable* from a rule set R if it is directly triggering reachable from R or if there is a rule r_t , which is triggering reachable from R , such that $\langle r_t, r_j \rangle \in TG$.

Definition 2. A rule r_j is *directly activation reachable* from a rule set R if there is a rule r_i in R such that $\langle r_i, r_j \rangle \in AG$; a rule r_j is *activation reachable* from a rule set R if it is directly activation reachable from R or if there is a rule r_a , which is activation reachable from R , such that $\langle r_a, r_j \rangle \in AG$.

Example 4. In Figure 1, r_2 is directly triggering reachable from $\{r_4, r_5\}$, r_3 is triggering reachable from $\{r_4, r_5\}$. In Figure 3, r_4 is directly activation reachable from $\{r_1, r_2, r_3\}$, r_5 and r_6 are activation reachable from $\{r_1, r_2, r_3\}$.

Definition 3. A rule r_i is a *triggering rule* of a rule r_j if there is a arc $\langle r_i, r_j \rangle \in TG$; A rule r_i is an *activation rule* of a rule r_j if there is a arc $\langle r_i, r_j \rangle \in AG$.

Definition 4. In TG, a path $\langle r_0, \dots, r_n, r_0 \rangle$ is called as a *TG cycle*. In AG, a path $\langle r_0, \dots, r_n, r_0 \rangle$ is called as an *AG cycle*.

Theorem 1. In the AG, if a rule r isn't in an AG cycle and isn't activation reachable from an AG cycle, r must be finitely executed.

Proof. It is intuitive that r 's condition finally is false by its self-deactivation. Thus, the conclusion holds.

Such notions as activation path and activation path set are presented based on Theorem1 in order to accurately express the infinite activation relationship.

Definition 5. If r_a is an activation rule of r , an *activation path* P_a associated with $\langle r_a, r \rangle \in AG$ can be defined as follows:

1. If r_a is in a AG cycle R_a , P_a is denoted as R_a ;
2. If r_a is not in a AG cycle R_a but it can be activation reachable from R_a , P_a is denoted as $R_a \cup p'$ where p' is such a path holding the following two properties:
 - (1) r_a is activation reachable from R_a along p' and p' does not contain any rule in R_a ;
 - (2) p' is minimal, i.e., if any rule is removed from p' then property (1) cannot hold.

Example 5. In Figure 3, for any rule in $\{r_1, r_2, r_3, r_4, r_5\}$, its activation path is the AG cycle R'_1 . For the rule r_6 , its activation path is (R'_1, r_5) .

Definition 6. A rule r 's *activation path set* consists of all activation paths with respect to its activation rules. It is denoted as $Path-Set_{act}(r)$.

Theorem 2. All paths in AG, which can infinitely activate r , must be contained by the r 's activation path set.

Proof. It is intuitively followed by theorem1 and definition5.

5 Analysis About Termination of a TG Cycle

5.1 Considering Synchronous Execution of an Activation Path

Definition 7. An *Irreducible Active Rule Set* is a subset of the active rule set R obtained by applying to R the Rule Reduction Algorithm in [2,3].

[2,3] have proved that all rules removed from an Irreducible Active Rule Set must be finitely executed. If no specific notice, all rules, TG cycles and AG cycles referred by following theorems are all in an irreducible rule set R .

Theorem 3. Let r be an arbitrary rule in a TG cycle R_T . It is followed that there always is an activation path P_a of r and any rule contained by P_a is in R_T . Thus, R_T may be non-termination.

Obviously, theorem3 depicts a very simple case and its conclusion is intuitive.

Theorem 4. Let r be an arbitrary rule in a TG cycle R_T . It is followed that there always is an activation path P_a of r and P_a holds one of the following two properties:

- (1) Any rule contained by P_a is in R_T ;
- (2) Besides some rules in R_T , the rest of rules contained by P_a aren't in R_T but they can be triggering reachable from R_T .

Thus, R_T may be non-termination.

If P_a holds the property (1), the conclusion obviously holds by theorem3. If P_a holds the property (2), assuming that a rule r' contained by P_a isn't in R_T , but a path, along which r' can be triggering reachable from R_T , always can be contained in an execution sequence of R_T . This can be done using the method [3] which exploits priorities of rules and dominant priority of a reaching set presented by [3]. That is, an execution sequence of R_T can contain P_a . So, r may be infinitely triggered and activated in an execution sequence of R_T , Theorem4 holds.

Theorem 5. Let r be a rule in a TG cycle R_T of an irreducible rule set R . P_a is an arbitrary activation path of r , which holds the following property: besides some rules that are in R_T or aren't in R_T but can be triggering reachable from R_T , the rest of rules contained by P_a cannot be triggering reachable from R_T . Thus, the following conclusions can be obtained:

- (1) Those rules contained by P_a , which cannot be triggering reachable from R_T , must be in a TG cycle R'_T or can be triggering reachable from R'_T .
- (2) Let r' be an arbitrary rule in R'_T , r' cannot be triggering reachable from R_T .
- (3) If R'_T is termination, R_T must be termination.
- (4) If R_T is non-termination, R'_T must be non-termination.
- (5) If R_T is always considered terminable, Termination decision made for R can be correct and efficient.

Proof. (a) Let r' be an arbitrary rule of P_a which cannot be triggering reachable from R_T . Since R is an irreducible rule set, r' must be in a TG cycle R'_T or can be triggering reachable from R'_T . Otherwise, r' must be removed from R by the Rule Reduction Algorithm [2,3]. So conclusion (1) holds.

(b) The conclusion (2) is proven by contradiction. Let r' be an arbitrary rule in R'_T and assuming that r' can be triggering reachable from R_T . According to the notion of triggering reachable, any rule, which is in R'_T or can be triggering reachable from R'_T , must be triggering reachable from R_T . This is in contradiction with the conclusion (1).

By the assumption of theorem5, there must exist four cases as follows.

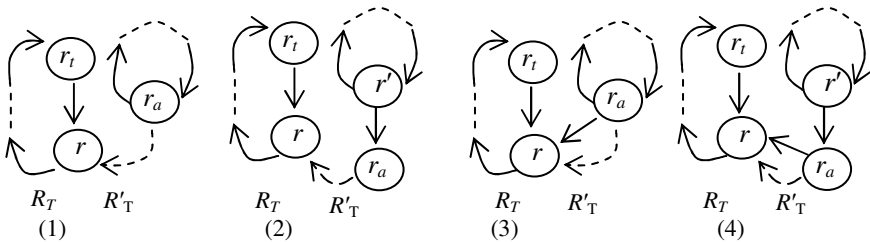


Fig. 6. TG and AG graphs associated with R_T in Theorem 5

(c) In Figure6, r_t is the triggering rule of r and r_a is the activation rule of r . 1) In Figure6 (1) and (2), r cannot be reachable from both r_t and r_a at the same time because R_T and R'_T cannot be synchronously executed. So r must be finitely executed. 2) In Figure6 (3) and (4), r_a can trigger r . This is the worst case. As R'_T is termination, r_a must be finitely executed. So r can be finitely activated by r_a . That is, r must be finitely executed. By 1) and 2), R_T must be termination. The conclusion (3) holds.

(d) In the worst cases illustrated by Figure6 (3) and (4), R_T may be non-termination. Assuming that R_T is non-termination and R'_T is termination, it is in contradiction with the conclusion (3). Therefore, the conclusion (4) holds.

(e) If R'_T is termination, R_T is deduced as terminable by the conclusion (3). But if R_T is termination, we should determine whether R'_T is termination to decide the termination of R . Thus, we should firstly consider the termination of R'_T . By the conclusions (3) and (4), R_T can be always considered as terminable when deciding the termination of R . Moreover, if R'_T can be guaranteed to be termination, R_T must be termination at the same time. So the conclusion (5) holds.

Regarding a TG cycle, which satisfies the condition of theorem5, as terminable, a following conclusion can be made from theorem3-5.

Corollary 1. Let r be an arbitrary rule in a TG cycle R_T . It is followed that there always is an activation path P_a of r and any rule contained by P_a can be triggering reachable from R_T . Thus, R_T may be non-termination. Otherwise, R_T must be termination.

5.2 The Formula Constructed for an Activation Path

A TG cycle, which may be non-termination judged by corollary1, can be termination detected by the technique of constructing a formula for an activation path.

Definition 8. A *rule execution sequence* (RES) of a TG cycle R_T is denoted by $\ll r_1, \dots, r_n \gg^+$ where r_i 's are all distinct. It indicates the cyclic execution from r_1 to r_n when a user transaction triggers at least one rule in R_T . All RESs belong to R_T form a set denoted as RES-Set (R_T).

Example 6. In Figure 3, there are two RESs of the TG cycle R_1 : $RES1 = \ll r_1, r_2, r_3 \gg^+$, $RES2 = \ll r_1, r_2, r_4, r_5, r_6, r_3 \gg^+$, $RES\text{-Set}(R_1) = \{RES1, RES2\}$.

Definition 9. Let r be a rule in a TG cycle R_T and P_a be an activation path of r . All rules in R_T form a rule set denoted as *Rules-Set* (R_T) and all rules contained by P_a form a rule set denoted as *Rules-Set* (P_a). δ is a RES in $RES\text{-Set}(R_T)$, δ *contains* P_a if $Rules\text{-Set}(\delta) \supseteq Rules\text{-Set}(P_a)$.

From corollary1, it can be deduced that any rule in a non-termination TG cycle R_T must have an activation path that can be contained by a RES of R_T . [11] provides a technique of constructing the formula for a path or a RES of R_T .

Definition 10. A path P is a *satisfiable path*, if $formula(P) = true$. Otherwise, P is a *non-satisfiable path*. A path P is a *key non-satisfiable path* with respect to a rule set R , if P is a non-satisfiable path and the end node of P belongs to R .

Let δ be a RES of a TG cycle R_T . If $Formula(\delta) = false$, there must be a non-satisfiable path ρ contained by δ . However, ρ cannot be sure to make the rules in $Rules-Set(R_T) \cup Rules-Set(P_a)$ terminable. So we present the notion of *key non-satisfiable path* with respect to a rule set in order to guarantee making a rule in R_T or P_a terminable. Thus, R_T or P_a must be termination when δ is performed.

Theorem 6. Let r be a rule in a TG cycle R_T and P_a be an activation path of r . If $\exists \rho \in RES - Set(R_T)$ such that ρ contains P_a and $Formula(\rho) \neq false$, then P_a may be infinitely executed with ρ . Otherwise, if it is followed that for any RES δ in $RES-Set(R_T)$, which contains P_a , it must have a key non-satisfiable path with respect to $Rules-Set(R_T) \cup Rules-Set(P_a)$, P_a must be finitely executed when any RES of R_T is performed by the system.

Example 7. In Figure 3, there are two RESs of R_1 as shown in Example6. r_6 has the only activation path P_a that is (R'_1, r_5) , $Rules-Set(P_a) = \{r_1, r_2, r_4, r_3, r_5\}$. Since $Rules-Set(RES2) \supseteq Rules-Set(P_a)$, $RES2$ contains P_a but $RES1$ doesn't. $Formula(RES2) = (T.M=1) \text{ AND } (T.M=0) = false$ and $\langle r_2, r_4 \rangle$ is a key non-satisfiable path with respect to $Rules-Set(R_1) \cup Rules-Set(P_a)$, so P_a must be finitely executed by theorem6. Thus, r_6 must be finitely activated. Assuming that $\langle r_5, r_6 \rangle$ is the only non-satisfiable path but $\langle r_2, r_4 \rangle$ isn't, at this time, $Formula(RES2) = false$. By [9-11], only $\langle r_5, r_6 \rangle$ can be removed from TG. So all rules in R_1 and P_a are left in TG and they may be infinitely executed.

5.3 Effective Activation Path

Definition 11. An AG cycle R_A is an *inhibited AG cycle*, if R_A is termination.

Theorem 7. Let R_A be an arbitrary AG cycle in an irreducible rule set R . If any rule in R_A can be triggering reachable from a TG cycle R_T which holds the following property: $\exists \rho \in RES - Set(R_T)$ satisfies $Rules - Set(\rho) \supseteq Rules - Set(R_A)$ and there is no key non-satisfiable path with respect to $Rules-Set(R_T) \cup Rules-Set(R_A)$, R_A may be non-termination. Otherwise, R_A must be termination.

Example 8. In Figure 3, Example7 has shown that only $RES2$ of the TG cycle R_1 contains the AG cycle R'_1 but $RES1$ doesn't. $\langle r_2, r_4 \rangle$ is a key non-satisfiable path, so $RES2$ should be only finitely executed. Thus, R'_1 must be termination.

Definition 12. An activation rule r is an *inhibited activation rule*, if r is only finitely executed.

Theorem 8. Let r be an arbitrary rule and isn't in an AG cycle. If r can be activation reachable from a AG cycle R_A and both r and any rule in R_A can be triggering reachable from a TG cycle R_T which holds the following property: $\exists \rho \in RES - Set(R_T)$ satisfies $Rules - Set(\rho) \supseteq \{r\} \cup Rules - Set(R_A)$ and there is no key non-satisfiable path with respect to $Rules-Set(R_T) \cup Rules-Set(R_A) \cup \{r\}$, r may be infinitely executed. Otherwise, r must be finitely executed.

Definition 13. An activation path without any inhibited AG cycle and any inhibited activation rule is called as an *effective activation path*.

Theorem 9. Let r be an arbitrary rule in a TG cycle R_T . It is followed that there always is an effective activation path P_a of r and any rule contained by P_a can be triggering reachable from R_T which holds the following property: $\exists \rho \in RES - Set(R_T)$ satisfies $Rules - Set(\rho) \supseteq Rules - Set(P_a)$ and there is no key non-satisfiable path with respect to $Rules - Set(R_T) \cup Rules - Set(P_a)$. R_T may be non-termination. Otherwise, R_T must be termination.

6 Termination Algorithm

Let R be an arbitrary irreducible rule set and $S_T = \{ R_T \mid R_T \text{ is a TG cycle in } R \}$ and $S_A = \{ R_A \mid R_A \text{ is an AG cycle in } R \}$.

Algorithm1. Refined Termination-test

Input: The TG cycle set S_T and the AG cycle set S_A

Output: If R is termination then return (true); otherwise, return (false)

Begin

- (1) **FOR** each AG cycle $R_A \in S_A$
 - flag: =true; /*Assuming that R_A is inhibited*/
 - IF** ($\exists R_T \in S_T$ satisfies that for $\forall r \in Rules - Set(R_A)$, r can be triggering reachable from R_T)
 - IF** ($\exists \rho \in RES - Set(R_T)$ satisfies $Rules - Set(\rho) \supseteq Rules - Set(R_A)$ and there is no key non-satisfiable path with respect to $Rules - Set(R_T) \cup Rules - Set(R_A)$)
 - flag: =false;
 - ENDIF**
 - ENDIF**
 - IF** flag
 - R_A is marked as inhibited;
 - ENDIF**
- ENDFOR**
- (2) **FOR** each rule r which isn't in an AG cycle
 - flag: =true; /*Assuming that r is inhibited*/
 - IF** ($\exists R_A \in S_A$ satisfies that R_A isn't inhibited and r can be activation reachable from R_A)
 - IF** ($\exists R_T \in S_T$ satisfies that for $\forall r \in Rules - Set(R_A)$, r and r can be triggering reachable from R_T)
 - IF** ($\exists \rho \in RES - Set(R_T)$ satisfies $Rules - Set(\rho) \supseteq \{r\} \cup Rules - Set(R_A)$ and there is no key non-satisfiable path with respect to $Rules - Set(R_T) \cup Rules - Set(R_A) \cup \{r\}$)
 - flag: =false;
 - ENDIF**
 - ENDIF**
 - ENDIF**
 - IF** flag
 - r is marked as inhibited;
 - ENDIF**
- ENDFOR**

```

(3) FOR each TG cycle  $R_p \in S_p$ 
    Sign: =false; /*Assuming that  $R_p$  isn't termination*/
    FOR each rule  $r \in Rules-Set(R_p)$ 
        flag: =true; /*Assuming that  $r$  is termination*/
        IF ( $\exists$  an effective activation path  $P_a \in Path-Set_{act}(r)$  sat-
            isfies that for  $\forall r \in Rules-Set(P_a)$ ,  $r$  can be trig-
            gering reachable from  $R_p$ )
            IF ( $\exists \rho \in RES-Set(R_p)$  satisfies  $Rules-Set(\rho) \supseteq Rules-Set(P_a)$  and
                there is no key non-satisfiable path with respect
                to  $Rules-Set(R_p) \cup Rules-Set(P_a)$ )
                flag: =false; /* $r$  can be infinitely executed*/
            ENDIF
        ENDIF
        IF flag
            Sign: =true; break; /* $R_p$  is termination*/
        ENDIF
    ENDFOR
    IF NOT(Sign)
        Return(false); /* $R$  is non-termination*/
    ENDIF
ENDFOR
(4) Return(true); /*Since no TG cycle is non-termination,  $R$  is termi-
    nation*/
END.

```

Theorem 10. Algorithm1. is correct and will terminate.

Proof. Algorithm1 is guaranteed to be correct by theorem7- 9. For TG cycles, RESs of a TG cycle, AG cycles, rules, activation paths in R , the numbers of them are finite, so algorithm1 can automatically terminate.

Example 9. We analyze the motivating Example2 through the algorithm1.

1. In Figure 3, R_1 has two RESs as shown in Example6. Since $Rules-Set(RES2) \supseteq Rules-Set(R'_1)$, $RES2$ contains R'_1 but $RES1$ doesn't. $\langle r_2, r_4 \rangle$ is a key non-satisfiable path with respect to $Rules-Set(R_1) \cup Rules-Set(R'_1)$, so R'_1 is inhibited.

2. As R'_1 is the only AG cycle in this rule set and it is inhibited, r_5 and r_6 can only be activation reachable from R'_1 , so they are both inhibited.

3. For the TG cycle R_1 , any rule in it has only one activation path R'_1 . As R'_1 is inhibited, no rule has an effective activation path. Thus, R_1 is termination. Since R_1 is the only TG cycle, this rule set must be termination.

The rule set in Example3 must be termination deduced by the algorithm1. As we analyzed in section3, [2,3,9-11] cannot determine these termination cases.

7 Conclusion

We have described the idea of activation path and formula constructed for activation path. Moreover, we have proposed a new method to detect more termination situations than previous works [2,3,9-11]. Since our methods are based on TG and AG, they are practical and general with respect to various rule languages from ECA language for XML to the trigger language in SQL: 1999 standard.

References

1. A.Aiken, J.Widom, J.Hellerstein. Behavior of Database Productions Rules: Termination, Confluence, and Observable Determinism. In *Proc. Int'l Conf. On Management of Data (SIGMOD)*, San Diego, California, 1992.
2. E.Baralis, S.Ceri, et al. Compile-Time and Runtime Analysis of Active Behaviors. *IEEE Transactions on Knowledge and Data Engineering*, 10(3)(1998) 353–370.
3. E.Baralis, S.Ceri, et al. Improved Rule Analysis by Means of Triggering and Activation Graphs. In *Proc. Int'l Second Workshop Rules in Database Systems (RIDS)*, Athens, Greece, 1995.
4. J.Bae, H.Bae et al. Automatic Control of Workflow Processes Using ECA Rules. *IEEE Transactions on Knowledge and Data Engineering*, 16(8)(2004) 1010–1018.
5. A.Bonifati, S.Ceri, and S.Paraboschi. Active rules for XML. A new paradigm for e-services. *VLDB Journal*, 10(1) (2001) 39–47.
6. J.Bailey, A.Poulovassilis, and P.T.Wood. An event-condition-action language for XML. In *Proc. WWW2002*, pages486–495, Hawaii, 2002.
7. J.Bailey, Guozhu Dong et al. On the Decidability of the Termination Problem of Active Database System. *Theor. Comput.Sci.*311 (1-3) (2004) 389–437.
8. A.Aiken, J.Hellerstein, J.Widom. Static Analysis Techniques for Predicting the Behavior of Database Production Rules. *ACM Transactions on Database Systems*, 20(1)(1995) 3–41.
9. A.P.Karadimce, S.D.Urban. Refined Triggering Graph: A Logic-Based Approach to Termination Analysis in an Active Object-Oriented Database. In *Proc. Int'l Conf. On Data Engineering (ICDE)*, New-Orlean, Louisiana, 1996.
10. S.Y. Lee, T.W. Ling. Refined Termination Decision in Active Databases. In *Proc.Int'l Conf. On Database and Expert Systems Applications (DEXA)*, Toulouse, France, 1997.
11. S.Y. Lee, T.W. Ling. A path Removing Technique for Detecting Trigger Termination. In *Proc. Int'l Conf. On Extended Database Technology (EDBT)*, Valencia, Spain, 1998.
12. E.Baralis, J.Widom. An Algebraic Approach to Static Analysis of Active Database Rules. *ACM Transactions on Database Systems*, 25(3)(2000) 269–332.
13. S.Comai, L.Tanca. Termination and Confluence by Rule Prioritization. *IEEE Transactions on Knowledge and Data Engineering*, 15(2)(2003) 257–270.
14. J.Bailey and A.Poulovassilis. Abstract interpretation framework for termination analysis in functional active databases. *Journal of Intelligent Information Systems*, 12(2/3)(1999) 243–273.
15. N.W.Paton, et al. Active Database System. *ACM Computing Surveys*, 31(1)(1999) 63–103.
16. S.Ceri, P.Fraternali, et al. Active Rule Management in Chimera.In S.Ceri, J.Widom, "Active Database Systems", Morgan Kaufmann, 1996.
17. L.Leverenz, et al. SQL Reference, Handbook. Oracle Corporation, 2000.

Stream Operators for Querying Data Streams

Lisha Ma¹, Stratis D. Viglas², Meng Li¹, and Qian Li²

¹ School of Mathematical and Computer Sciences,
Heriot-Watt University, Edinburgh, United Kingdom

² School of Informatics, University of Edinburgh,
Edinburgh, United Kingdom

Abstract. One of the most important uses of aggregate queries over data streams is sampling. Typically, aggregation is performed over sliding windows where queries return new results whenever the window contents change, a concept referred to as a continuous query. Existing data models and query languages for streams are not capable of expressing many practical user-defined samplings over streams. To this end we propose a new data stream model, referred to as the *sequence model*, and a query language for specifying aggregate queries over data streams. We show that the sequence model can readily express a superset of the aggregate queries expressible in the previously proposed time-based data stream model, thus providing a declarative and formal semantics to understand and reason about continuous aggregate queries. Defined on top of the sequence model, our query language supports existing sliding window operators and a novel *frequency operator*. By using the frequency operator one is capable of expressing useful sampling queries, such as queries with user-defined group-based sampling and nested aggregation over either the input stream or the result stream. Such capabilities are beyond those of previously proposed query languages over streams. Finally, we conduct a preliminary experimental study that shows our language is effective and efficient in practice.

1 Introduction

The proliferation of the Internet, Web technology, and sensor networks highlight the need for treating data as a continuous data stream. One major challenge is the development of techniques for providing continuously updated answers to aggregate queries over potentially unbounded streams. A general approach for addressing this challenge is by means of windows queries, which add window clauses to continuous queries and thus allow aggregate queries to be evaluated over a segment of the input data stream rather than over the entire stream. There has been a great deal of research on developing algorithms for windowed aggregate queries [2,5,6,9,10,12,13].

However, current techniques are limited in two crucial aspects. Firstly, most query languages over streams do not have the necessary constructs to support *condensative* query processing over streams, as is the case in traditional DBMSs. We call aggregate query processing in a traditional DBMS *condensative*, since the answer to an aggregate query is usually much smaller than the answer to its non-aggregate counterpart. Having a declarative stream language that supports condensative query processing is crucial

since data streams are potentially unbounded, while main memory and secondary storage have fixed limits; new query results should therefore be generated whenever the window contents change. Secondly, the focus of previous work has mainly been on query evaluation while fundamental questions in connection with data models and formal semantics for queries have not yet been thoroughly addressed. This lack makes it difficult to reason about aggregate queries and compare different languages in a uniform semantics. The situation is aggravated when one moves to the realm of distributed computation, as is usually the case when dealing with data streams.

Scenario and Example Queries. In practice there has been an increasing need for aggregate queries. To illustrate this, consider a scenario of Grid monitoring where one wants to know the current state of the Grid and its status in the past. Suppose we have a core relation `cpu` that publishes the results of monitoring CPU workload. The relation has the following schema: `cpu(id, site, load, timestamp)`, where `id` denotes a unique machine identifier, `site` denotes where the machine is located, and `load` records the current CPU load of the machine. Consider the following queries:

Aggregate query: For every 1000 new measurements and for each node (site) N in the Grid, and for machines located in N in the last hour, how many of those machines currently have a CPU load less than 30%?

Nested aggregate query: On a per-hour basis and for machines located at Heriot-Watt University, what is the average number of machines that had a CPU load less than 30% during the last hour? The previous aggregate query could be used to evaluate the nested aggregate one.

The above queries demand “jumping” between two different granularities: a tuple-based granularity and a time-based granularity, respectively. We term such a query a *mixed jumping window* query. In addition, the second query requires nested aggregation to be computed w.r.t the nested jumping window. Note that neither the current stream data model, nor window aggregate operators support such queries — not even in the case of a centralized DSMS (Data Stream Management System).

Prior Work. To the best of our knowledge, the most expressive language for specifying window queries is StreaQuel, implemented in TelegraphCQ [7]. In StreaQuel, each query is defined in terms of a for-loop construct that specifies (1) the set of windows over which the query is to be executed, and (2) how often the query should be run.

Recently Li *et al.* [13] have proposed a declarative way of defining windowed aggregate queries. Their window definition has three parameters: `RANGE` specifies the window size, `SLIDE` the window movement, and `WATTR` the granularity, *i.e.*, whether `RANGE` and `SLIDE` are defined in terms of timestamps or sequence numbers of tuples. All these patterns were defined with respect to window identifiers. Such semantics define a function to uniquely identify each window extent for a given window aggregate query; also, they require an inverse function that, for each tuple, it determines the extents of the window to which the tuple belongs. Window identifier semantics was implemented in an extended version of the Niagara Query Engine [8] for evaluating aggregate window queries over data streams. Note that none of those approaches can express the example queries given above. The reasons are that (a) frequency and window length need to be defined in terms of the same granularity, and (b) the frequency over the input stream cannot be differentiated from the frequency over the result stream.

A host of research exists on tackling aggregate query evaluation over data streams. Widely differing approaches employing, *e.g.*, hashing, sampling, sketches and wavelets, just to name a few, have been proposed in the literature [4,11,12,14]. All those approaches, however, are workload-driven, as opposed to being user-driven. In more detail, it is desirable to give the user considerable freedom in how windows are defined and handled by the system. In other words, the user should be in a position to declare window semantics at the language level, rather than rely on the system to deduce trade-offs between accuracy and resource consumption, as is usually the case. This is what we term *condensative query evaluation*: the user should be able to merely *declare* how frequently sampling should take place and the system should perform aggregation based on the user’s instruction.

Contribution and Overview. In order to make existing stream query languages more expressive and support user-driven condensative processing over data streams, we introduce a new model, referred to as the *sequence model*, and novel operators, termed *frequency operators*. Our model and stream operators have a solid theoretical foundation, and are capable of expressing previous models and operators. As a result they provide cleaner semantics, allowing us to better understand and reason about aggregate queries over streams. More specifically, our main contributions are the following:

- We present a formal semantics that models time-varying stream data as a function over an ordered sequence domain; this is called a *sequence model*. Our semantics draws features from existing work on sequence databases [17] and the time-based approach of modeling stream data [1]. It is capable of expressing complex queries in an intuitive manner.
- We introduce a novel operator, termed the *frequency operator*, along with its formal semantics. We show that our query language can specify window queries found in [1,5,7,8,18]. Moreover, the query language provides functionality to support queries with user-defined group-based sampling and various forms of aggregation.
- Based on the previous two ideas, we have implemented our conceptual operators in a prototype query engine. To demonstrate the efficiency gained by pushing the frequency operator towards the base streams for jumping window queries over an ordered sequence stream, we compare it with the approach based on window identifiers. Our preliminary experimental results show that a pushed down frequency operator is effective and outperforms the window identifier approach. We further show how to evaluate “mixed jumping window” queries, something which cannot be addressed by existing approaches.

Organization. Section 2 reviews sequence databases and the time-based data model. Sections 3 and 4 introduce the formal semantics of our language. Section 5 presents our experimental results, while Section 6 concludes the paper.

2 Previous Approaches

In this section we review two previous data models from which our model draws some features.

2.1 Sequence Database Model

The SEQ sequence model and algebra were introduced by Seshadri *et al.* in [17], which defines sequences as an ordering function from the set of integers to each item in the sequence. The SEQ model separates the data from the ordering information, and it can deal with different types of sequence data by supporting an expressive range of sequence queries.

Most manipulation operators, such as selection, projection, various set operations, and aggregation (including moving windows) are carried over from the relational data model. There are also many new operators developed specifically for manipulating sequences. The SEQ model has been implemented in SRQL (Sorted Relational Query Language) [16].

2.2 Time-Based Stream Model

In a streaming environment, data items appear in a time-varying, continuously arriving, and append-only fashion. In order to capture its constantly changing essence, a formal semantics, in the form of the *time-based stream model*, and the Continuous Query Language [3] (CQL) were introduced in [1]; CQL has been implemented in the STREAM system [15]. Continuous queries can be registered against a data stream \mathcal{S} or relation \mathcal{R} . More formally, one can give the precise semantics for streams and relations as follows: Let \mathcal{D}_R be the set of all tuples that satisfy the schema R . Let \mathcal{T} be the set of all timestamps. Then, a stream \mathcal{S} with schema R is a subset $\mathcal{S} \subseteq \mathcal{D}_R \times \mathcal{T}$, such that for every $\tau \in \mathcal{T}$ the bag $\{e \mid \langle e, \tau \rangle \in \mathcal{S}\}$ is finite. With $\mathcal{P}(\mathcal{D}_R)$ we denote the set of all subsets of \mathcal{D}_R . A time-dependent relation \mathcal{R} for the schema R is a mapping $\mathcal{R}: \mathcal{T} \rightarrow \mathcal{P}(\mathcal{D}_R)$, such that each set $\mathcal{R}(\tau)$ is finite. With these definitions, we can have transformation from a stream to a relation w.r.t. time.

3 Sequence Model

In a time-based model the order of tuples is not uniquely defined. This drawback leads to ambiguous semantics for continuous queries involving tuple-based sliding windows or moving aggregation. To address this issues, we draw features from sequence databases and construct a sequence-based model for streams. In this section we introduce our model and a well-defined formal semantics. As will be seen, our model is more expressive than the time-based model. An abstract *relational stream* has the following characteristics:

- A stream consists of tuples;
- A stream has a relational schema and all its tuples adhere to that schema;
- A stream develops over time. Therefore, it is assumed that there is a set \mathcal{T} to represent the time domain. A *timestamp* is any value from \mathcal{T} ; timestamps are linearly ordered.

Relational Schema. A relational schema has the form:

$$R\langle a_1 : T_1, \dots, a_k : T_k, \text{timestamp} : \text{Date} \rangle,$$

where R is a relation symbol, a_1, \dots, a_k are attributes, and T_1, \dots, T_k are types as in SQL. In the time-based model, the timestamp is an optional attribute. It is used only if there is a need to separate the generation time and arrival times of a tuple. In the sequence model, however, we say that a timestamp attribute should be present by default and it records a tuple's arrival time.

Time Domain. Although there is no restriction on whether the time domain is modeled after real clock time or natural numbers, it is still required to define its general properties. A time domain should be *ordered*. Let $\sqsubseteq: X \times X$ be an ordering (*i.e.*, \sqsubseteq is reflexive, antisymmetric and transitive). For every ordering \sqsubseteq we define the strict version \sqsubseteq' of \sqsubseteq by $x \sqsubseteq' y$ iff $x \sqsubseteq y$ and $x \neq y$. A binary relation is:

Linear: if for all $x, y \in X$ we have either $x \sqsubseteq y$ or $y \sqsubseteq x$.

Dense: if for all $x, y \in X$ where $x \sqsubseteq' y$, there exists a $z \in X$ such that $x \sqsubseteq' z$ and $z \sqsubseteq' y$.

Discrete: if for every two elements $x, y \in X$ there are only a finite number of elements z between them, *i.e.*, there are only finitely many z such that $x \sqsubseteq z$ and $z \sqsubseteq y$.

If a linear ordering is discrete, then for every element $x \in X$, there is either a *least* element that is greater than x , or there is no element greater than x . We specify that a time ordering should have following properties:

Definition 1 (Time Domain Properties). (1) Any two timestamps must be comparable, which means that the ordering is linear. (2) The ordering should not be dense, but discrete.

A time domain with these properties is essentially identical with either the set of all integers or an integer interval. This also allows us to define the length of a sliding window. A window of length n consists of the starting point plus the next $(n - 1)$ elements. An unbounded window can be modeled by only specifying its starting point.

Given the above semantics, the sequence model is defined as follows: Let \mathcal{D}_R be the set of all tuples that satisfy relational schema R . By default, every relational schema has an attribute τ , called the *timestamp* attribute, which is of type DATETIME. We use \mathbb{N} to denote the set of natural numbers and model a stream \mathcal{S} for a relation \mathcal{R} as a partial function from the natural numbers to \mathcal{D}_R .

Definition 2 (Sequence Stream). A stream for a relation \mathcal{R} is a partial function $\mathcal{S}: \mathbb{N} \hookrightarrow \mathcal{D}_R$, such that whenever $\mathcal{S}(n)$ is defined for some $n \in \mathbb{N}$, then $\mathcal{S}(m)$ is defined for every $m < n$, as well. For every stream tuple $\mathcal{S}(n)$ we denote the value of its timestamp attribute as $\mathcal{S}^\tau(n)$.

According to the definition, a stream \mathcal{S} is either defined for all numbers, or there is a number $n \in \mathbb{N}$ such that $\mathcal{S}(n)$ is defined and $\mathcal{S}(m)$ is undefined for all $m > n$. A special case is the empty stream, denoted as \perp , which is undefined for every $n \in \mathbb{N}$. Observe that our definition does not make any assumption as to whether the timestamps of tuples and the order in which tuples arrive are compatible. However, some stream operators only make sense over streams where the tuples arrive in the order of their timestamps.

Definition 3 (Ordered Stream). A stream \mathcal{S} is ordered if for all $m, n \in \mathbb{N}$ we have that $\mathcal{S}^\tau(m) \leq \mathcal{S}^\tau(n)$ whenever $m \leq n$.

In this way, we allow different tuples to have the same timestamp, but they will still be ordered among them. Note that our model does not make assumptions about how often tuples arrive.

4 Stream Operators

In this section we shall use our model to define the existing sliding window operators, as well as the novel frequency operator. We show that queries expressible in the time-based model can also be specified in the sequence model. Furthermore, as will be seen in Section 4.5, it is possible to use the sequence model and associated operators to express queries that go beyond the capabilities of previous models and query languages.

4.1 Sliding Window Operators

We express the bound of the sliding window either as W_n for a tuple-based sliding window or W_t for a time-based window.

(1) **Tuple-based Sliding Window:** A tuple-based sliding window will slide whenever a new tuple arrives. For every $n \in \mathbb{N}$, we have a tuple-based sliding window W_n over stream \mathcal{S} , which produces a sequence of sets $W_n\mathcal{S}(j)$:

$$W_n\mathcal{S}(j) = \{\mathcal{S}(k) \mid k \leq j \text{ and } k \geq \max\{0, j - n\}\}.$$

(2) **Time-based Sliding Window:** A time-based sliding window W_t is bounded *only* by time t , while we do not know how many tuples are exactly within the window. However, it slides as time progresses. The sliding rate obviously depends on the time granularity. More formally, we define the output stream $W_t\mathcal{S}$ of a time-based sliding window as a sequence of sets $W_t\mathcal{S}(j)$ for a given j in stream \mathcal{S} . We say $W_t\mathcal{S}(j)$ is not defined if $\mathcal{S}(j)$ is not defined, otherwise we have

$$W_t\mathcal{S}(j) = \{\mathcal{S}(k) \mid k \leq j \text{ and } \mathcal{S}^\tau(k) + t \geq \mathcal{S}^\tau(j)\}.$$

4.2 Frequency Operator

The frequency operator F will pick the stream tuple based on a defined frequency. Depending on the unit used to set the frequency (*i.e.*, number of tuples, or time) we can have either (a) the tuple-based frequency operator F_n , or (b) the time-based frequency operator F_t respectively.

(1) **Tuple-based Frequency Operator:** For every natural number $n \in \mathbb{N}$ we have a tuple-based frequency operator F_n , which selects every n -th tuple of a stream. Formally:

$$F_n\mathcal{S}(j) = \mathcal{S}(n \times j).$$

(2) **Time-based Frequency Operator:** For every time instance t , we have the time-based frequency operator F_t , which selects tuples with timestamp $j \times t$ as a new stream $F_t\mathcal{S}$, where $j \in \mathbb{N}$. If there is no tuple with timestamp $j \times t$, then we will output the last tuple within that time slot. We say $F_t\mathcal{S}(j)$ is a subsequence of tuples with order j over order n_j of stream \mathcal{S} , where $j \in \mathbb{N}$. Then let

$$n_j = \max \{k \in \mathbb{N} \mid (j - 1) \times t \leq \mathcal{S}^\tau(k) \leq j \times t\}, \text{ if } \mathcal{S}(n_j) \neq \emptyset,$$

otherwise it is undefined. Now, $F_t\mathcal{S}(j) = \mathcal{S}(n_j)$, for all $j \in \mathbb{N}$ if n_j is defined, otherwise $F_t\mathcal{S}(j) = \perp$.

4.3 Jumping Window

Sometimes, we want our window to move faster than the rate at which it is sliding. Instead of posing the frequency operator over a sequence of tuples, we can also pose the frequency operator over a sequence of sets. We term such kind of a window a *jumping window*. Depending on how we set the frequency length, we classify the jumping window into two different types: tuple-based jumping windows and time-based jumping windows.

(1) **Tuple-based Jumping Window:** For every number $n \in \mathbb{N}$, and a sequence of sets WS that are produced by the sliding window operators W_n or W_t , we have a tuple-based jumping window $F_n(WS)$, which selects every n -th set of WS as follows:

$$(F_n(WS))(j) = WS(n \times j).$$

(2) **Time-based Jumping Window:** For a sequence of sets WS that are produced by the sliding window operators, we have a time-based jumping window $F_t(WS)$ by selecting a subsequence of sets with index j over index n_j of WS w.r.t. every time instance t . We define $F_t(WS)(j)$ for an arbitrary stream \mathcal{S} and a window operator W_n or W_t recursively by saying what it is the set $F_t(WS)(j)$ for an arbitrary number j . We first define the set of indices I_1 as

$$I_1 = \{j \in \mathbb{N} \mid \exists k \mathcal{S}^\tau(k) = j \times t\}.$$

If $I_1 \neq \emptyset$, then let $n_1 = \min I_1$, and define $(F_t(WS))(1) := WS(n_1)$, otherwise let $F_t(WS) = \perp$. Now, suppose n_j is defined for some $j \in \mathbb{N}$. Then let

$$I_{j+1} = \{j \in \mathbb{N} \mid \exists k \mathcal{S}^\tau(k) = j \times t \text{ and } k > n_j\}.$$

If $I_{j+1} \neq \emptyset$, then let $n_{j+1} = \min I_{j+1}$ and define $(F_t(WS))(j+1) := WS(n_{j+1})$, otherwise $(F_t(WS))(j+1) = \perp$. However, it is possible that $(F_t(WS))(j+1)$ is not defined because $\mathcal{S}^\tau(k) \neq j \times t$. For this case, we will output the latest set $WS(n_j)$ within time $[(j-1) \times t, j \times t]$ where

$$n_j = \max \{k \in \mathbb{N} \mid (j-1) \times t < \mathcal{S}^\tau(k) \leq j \times t\}.$$

4.4 Local and Non-local Semantics

A stream operator is a function Ω that takes a stream \mathcal{S} as input and outputs a stream $\Omega\mathcal{S}$. There should be a simple semantics to categorise all stream operators beyond the differences of the various stream data models. To this end, we introduce the semantics of *local* and *non-local* operator types for stream languages.

We first define a mapping from sequence model to time-based model. Let \mathcal{D}_R be the set of all tuples that satisfy the schema R . Let τ denotes the value of the timestamp attribute of the tuples and \mathcal{T} be the set of all existing timestamps. A stream \mathcal{S} in the sequence model can map to time-based model for all existing values of τ :

$$\mathcal{S}: \mathcal{T} \rightarrow \mathcal{D}_R,$$

Suppose we have an operator \mathcal{Q} that transfers a data stream \mathcal{S} from a sequence model to a time based model $\mathcal{Q}\mathcal{S}$, and $\mathcal{Q}\mathcal{S}(\tau)$ represents a bag of the tuples that have a timestamp equal to τ .

Definition 4. We say Ω is local if:

$$\mathcal{QS}_1(\tau) = \mathcal{QS}_2(\tau) \text{ then } (\Omega(\mathcal{QS}_1))(t) = (\Omega(\mathcal{QS}_2))(\tau),$$

otherwise it is non-local.

Such semantics provide the theoretical foundation of evaluating the operators in a uniform data model. Using it, we can easily define the semantics of the frequency operators in the time-based model. Note that as a time-based model is not aware of the order of tuples, we can only rewrite the semantics of our time-based operators within the time-based model.

Time-Based Frequency Operator. For every time instance t , we have a time-based frequency operator F_t . Conceptually, it selects sets with timestamp $(j \times t)$ as a new stream $F_t\mathcal{S}$. We then define $(F_t\mathcal{S})(\tau)$ for all $j \in \mathbb{N}$:

$$(F_t\mathcal{S})(\tau) = \{\mathcal{S}(\tau) \text{ if } \tau = j \times t \text{ for some } j \in \mathbb{N}\}.$$

Time-Based Jumping Operator. For every time instance t , and a stream WS that is produced by a sliding window operator, we have a time-based Jumping window $F_t(WS)$ obtained by selecting a bag for every t instance from stream WS . We then define $(F_t(WS))(\tau)$ for an arbitrary time t , where $t \in \mathcal{T}$ as:

$$(F_t(WS))(\tau) = \{WS(\tau) \text{ if } \tau = j \times t \text{ for some } j \in \mathbb{N}\}.$$

4.5 Example Queries

In a SQL-based language, a frequency operator can be expressed by adding to the range variable of a stream, say \mathcal{S} , the expression `[Frequency F]`, where F denotes an interval length. The length can be defined either in terms of number of tuples, as in `[Frequency n Tuples]` (i.e., “every n tuples”), or in terms of a time period, e.g., `[Frequency t Minutes]` (i.e., “every t minutes”). The operator picks tuples based on the predefined frequency length from the stream. To denote group-based sampling, we use `[Frequency F Partitioned By A_1, \dots, A_k]`. The operator partitions \mathcal{S} into different substreams based on the grouping attributes A_1, \dots, A_k ; then, for each substream, the operator picks tuples based on the predefined frequency. We separate the frequency over an input stream and a result stream by putting the frequency expression in either the `FROM` clause or the `SELECT` clause respectively.

We can combine the frequency operator with various sliding window operators to denote jumping windows. Saying `Frequency = 1 Tuple` is equivalent to a normal sliding window. Depending on how the frequency length is defined, we distinguish between *tuple-based* and *time-based* jumping windows. Instead of computing the answer whenever a new tuple arrives, the frequency operator requires a computation only after an interval of the frequency length. This means that, the operator will “sleep” between any two computations. A jumping window has two parameters:

The window size W . All tuples that arrive from the start during a period of length W , or the next W tuples have to be stored for computation.

The “sleep” length F . A new window is only output after the system sleeping for the designated period.

A jumping window is always defined by a sliding window operator followed by a frequency operator expression, as in [Range W , Frequency F]. Our semantics supports the mixing of tuple-based frequencies with time-based window bounds and vice versa. Such windows are called “mixed jumping windows”.

We next show our model and operators are more expressive than previous model and languages by giving the example queries into extended SQL with the proposed language construct.

Query 1. For every 1000 new measurements and for each node (site) N in the Grid, and for machines located in N in the last hour, how many of those machines currently have a CPU load less than 30%?

```
SELECT  site, COUNT(DISTINCT(id))
FROM    cpu[Range 1 Hour, Frequency 1000 Tuples]
WHERE   cpuload <= 30
GROUP BY site
```

This mixed jumping window query will count, for each site and after every 1000 tuples, the total number of computers that currently have a CPU load of less than 30% in the last hour’s worth of CPU tuples. It returns all the results obtained by evaluating the relational query over the window, at the end of each sleeping period.

Query 2. On a per-hour basis and for machines located at Heriot-Watt University, what is the average number of machines that had a CPU load less than 30% during the last hour? The previous aggregate query could be used to evaluate the nested aggregate one.

```
SELECT  AL.site, AVG(AL.number)
FROM    (SELECT  site, COUNT(DISTINCT(id))
        FROM    cpu[Range 1 Hour, Frequency 1000 Tuples]
        WHERE   cpuload <= 30
        GROUP BY site)
        AS avgNum(site, number) AL
[Range 1 Hour, Frequency 1 Hour]
GROUP BY AL.site
```

This query contains a nested frequency. Note that the nested frequency synchronizes the inner and the outer queries so as to avoid duplicate answers. Although the inner frequency is bounded in terms of tuples, it can still be controlled by the outer frequency (every minute). Finally, note that it is possible to register the inner query independently and use it as a primary source for answering the outer query.

5 Experimental Study

To evaluate the effectiveness of our semantics, we implemented our conceptual operators in a prototype query engine and conducted a preliminary experimental study. Our framework was implemented in Java and our experiments were executed on a Pentium IV 2.4Ghz with 512M of physical memory. We report wall clock timings and calculate execution time by measuring the average cost of 10000 answer tuples. Streaming behavior was simulated by using a pull-based execution model: the more effective the

algorithm, the more tuples it is able to process. A frequency operator typically spends its time sampling and aggregating, so there is a clear division of work. We are interested in showing how it is possible to optimize the sampling cost in such an environment, as we want to treat the efficiency of the aggregation algorithm as an orthogonal issue. Therefore, we used the same aggregation in all experiments, and have calculated the execution time as the sum of scanning the input stream and producing the aggregate. As a result, any performance gain we observe will be due to the efficiency of the sampling methodology, which is directly tied to how well the semantics of the operators can be implemented.

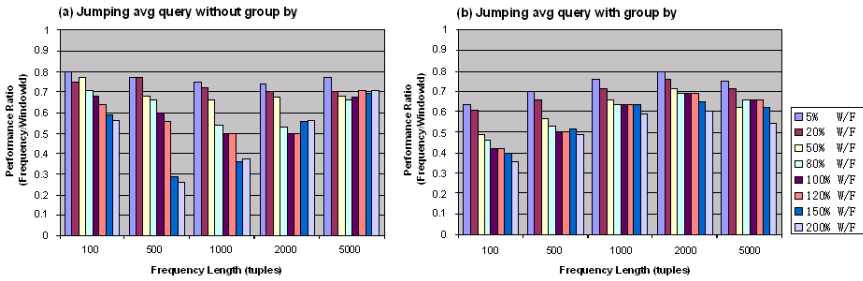


Fig. 1. Cost Ratio of Frequency vs. Window Id Approach

Experiments are divided into two parts. Firstly, we evaluate the performance of the pushed down frequency operators in contrast to the window identifier approach for a tuple-based jumping window (AVG) query. Note that we consider the case where one tuple may be in the contents of multiple windows. The efficiency of evaluating queries without or with a GROUP BY-clause is shown in Figures 1(a) and 1(b) respectively. The horizontal axis is the frequency length measured in tuples. The vertical axis is the performance ratio between the execution time using a pushed down frequency operator, over the execution time of the window identifier approach. The window length is represented as a percentage of the frequency length. For example a 30% W/F ration for a frequency length F of 1000 tuples will evaluate the query over a window length bounded by 300 tuples. As an independent operator, the frequency operator can be easily pushed down in a query plan to avoid unnecessary computation. This allows us to split aggregate query processing in two levels: (i) tuple sampling, and (ii) aggregation evaluation; this modeling provides a flexible mechanism to interact with different advanced aggregate operators. Our experiment showed that pushing down the frequency operator is an effective technique and it significantly outperforms the window identifier approach. Secondly, we evaluate the efficiency of processing mixed jumping window queries, which cannot be handled by existing approaches. Figure 2 shows the upper bound performance time of a mixed jumping window (AVG) query that has a frequency length specified on a tuple basis and a window length specified on a time basis. The horizontal axis is the frequency length (measured in tuples) which the window length is measured in seconds. The vertical axis is the total execution time (per answer tuple appearing in the average) measured in milliseconds. Note that performance can be further improved if a more efficient aggregation algorithm is employed.

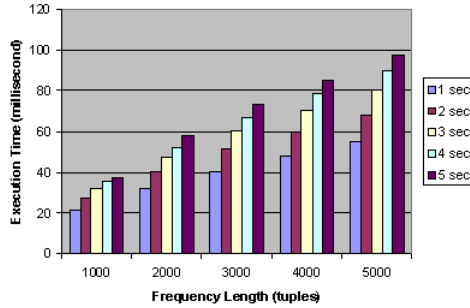


Fig. 2. Performance of Mixed Jumping Window

6 Conclusion

We have studied stream queries from a theoretical angle. More specifically, we have incorporated sampling in a declarative fashion to a query language. We have also introduced a formal semantics for both the new data model and proposed the frequency frequency operator for extending stream query languages with more expressibility, *i.e.*, allowing for user-defined sampling and condensative query processing. Our new frequency operator can be combined with the existing sliding window operators, a framework that yields the powerful “jumping window” operators. Those allow for mixed window specification (*i.e.*, both tuple-based and time-based) that can not be handled by existing query languages. We have tested the viability of the approach by performing an experimental study based on the optimization heuristic of pushing down such operators closer to the input streams. The preliminary results verify the effectiveness of the approach.

There are at least two avenues for future work. First, we are interested in integrating advanced aggregation algorithms with the proposed operators and providing a more thorough experimental study. Second, we would like to extend the semantics to address information integration scenarios that require the ability to automatically construct distributed query plans, based on techniques of answering continuous queries using continuous views. A key operation in creating such plans is determining whether a query is contained in another query. While this problem has been thoroughly investigated for queries over static databases, it is still open for continuous queries. Frequency operators may play an interesting role in this research, since they will give rise to further possibilities of containment. As the frequency operator is a declarative way of describing how to transform streams into smaller result streams, it will be useful for queries in large distributed applications.

Acknowledgments. The authors would like to thank Werner Nutt and Alasdair J. G. Gray for valuable discussions and comments. Werner Nutt played a significant role in shaping the main idea of the paper.

References

1. A. Arasu and J. Widom. A denotational semantics for continuous queries over streams and relations. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 33(3):6–11, 2004.
2. A. Arasu and J. Widom. Resource sharing in continuous sliding-window aggregates. In *VLDB*, pages 336–347, 2004.
3. Arvind Arasu *et al.* CQL: A Language for Continuous Queries over Streams and Relations. In *DBPL*, pages 1–19, 2003.
4. B. Babcock *et al.* Load shedding for aggregation queries over data streams. In *ICDE*, pages 350–361. IEEE Computer Society, 2004.
5. D. Carney *et al.* Monitoring streams - A new class of data management applications. In *VLDB*, pages 215–226, 2002.
6. S. Chandrasekaran and M.J. Franklin. Streaming queries over streaming data. In *VLDB*, pages 203–214, 2002.
7. S. Chandrasekaran *et al.* TelegraphCQ: Continuous dataflow processing. In *SIGMOD Conference*, pages 668–668, 2003.
8. J. Chen, D.J. DeWitt, F. Tian, and Y. Wang. Niagaracq: A scalable continuous query system for internet databases. In *SIGMOD Conference*, pages 379–390, 2000.
9. C. Cranor *et al.* Gigascope: A stream database for network applications. In *SIGMOD Conference*, pages 647–651, 2003.
10. A. Dobra, M.N. Garofalakis, J. Gehrke, and R. Rastogi. Processing complex aggregate queries over data streams. In *SIGMOD Conference*, pages 61–72, 2002.
11. J. Gehrke *et al.* On computing correlated aggregates over continual data streams. In *SIGMOD Conference*, 2001.
12. A.C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In *VLDB*, pages 79–88, 2001.
13. J. Li *et al.* Semantics and evaluation techniques for window aggregates in data streams. In *SIGMOD Conference*, 2005.
14. A. Manjhi *et al.* Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *SIGMOD Conference*, 2005.
15. Rajeev Motwani *et al.* Query Processing, Approximation, and Resource Management in a Data Stream Management System. In *CIDR*, 2003.
16. R. Ramakrishnan, D. Donjerkovic, A. Ranganathan, K.S. Beyer, and M. Krishnaprasad. SRQL: Sorted relational query language. In *SSDBM*, pages 84–95, 1998.
17. P. Seshadri *et al.* Seq: A model for sequence databases. In *ICDE*, pages 232–239, 1995.
18. Y. Yao and J.E. Gehrke. Query processing in sensor networks. In *CIDR*, 2003.

Join Algorithm Using Multiple Replicas in Data Grid*

Donghua Yang¹, Jianzhong Li^{1,2}, and Qaisar Rasool¹

¹ School of Computer Science and Technology, Harbin Institute of Technology, China

² School of Computer Science and Technology, Heilongjiang University, China
{yang.dh, lijzh, qrasool}@hit.edu.cn

Abstract. We address the issue of applying join operation to the relational data managed in the form of multiple replicas in data grid. A join algorithm is developed that solves the problem of replica matching and execution node selection by making use of matching theory. To ease the data transference among the grid nodes, we introduce relation reduction and row blocking techniques. The analytical and experimental results depict the effectiveness of the proposed join algorithm for the better network transfer rate and the efficient processing of replicated data to improve query responses in data grid.

1 Introduction

Data grid [1,2,3,4] is a distributed architecture for data management that provides the coordinated and collaborative mechanisms for integrating data distributed across network and forms a single, virtual environment for data access and management. The aim is to facilitate world-wide community with geographically-distributed resources for large-scale data-intensive applications and providing fast, reliable and transparent access to data. Many applications can be developed in data grid environment, such as most prominent being global climate simulation, high energy physics and biology computing.

Data replication [5] is an optimization technique well known in the distributed systems and database communities as a means of achieving better access times to data, fault tolerance and increasing the performance and robustness of distributed applications. Various replica management systems [6,7,8] exist, such as the Globus MDS-2 replica catalog, the Grid Data Mirroring Package (GDMP), the Giggie framework and the SDSC's Storage Resource Broker (SRB). These systems differ in several architectural aspects but allow consistent and locatable replicas, with a support of managed replicas' lifetime.

The roles which the database management systems are contributing towards data storage, access, management and transference in distributed environment [9,10,11] is tremendous. Among database operations, *Join* is a commonly used, but complex, operation that requires more time to complete than other operations and is an active area of attention [12,13]. Existence of autonomous nodes, massive and heterogeneous

* This work is supposed by the National Natural Science Foundation of China, Grant No. 60273082 and 60473075, the key Natural Science Foundation of Heilongjiang Province, Grant No. zjg03-05.

datasets and the different and unstable bandwidths among nodes brings new challenges to the Join operation in data grid environment.

In this paper, a join algorithm is proposed that facilitates the queries involving join operation to access the replicated *relational* data over the grid. Our work facilitates the users who issue the queries to access the data via grid that involves the join operation. Firstly a reduction algorithm is applied to the data involved. Second grid nodes, termed as *execution nodes*, are selected on which query (join operation) will be performed. The *reduced* data from the first step is then transferred over the grid to the execution nodes for processing and the results are propagated back to users' nodes. For convenience in data transference and data size we suggest new reducing and merging algorithms and sending techniques. The influence of various parameters on the performance of algorithm is studied by experiments, showing the efficiency and usefulness of the work.

The rest of the paper is organized as follows. Section 2 introduces algorithms for relation reduction, block transferring and merging. In Section 3 we discuss Edge-weight-minimum-matching algorithms by employing some important concepts from Graph Theory. Section 4 explores the proposed Join Algorithm in five phases. The results of experiments to measure the algorithm performance are given in Section 5. Finally we conclude the paper and highlight some future work in Section 6.

2 Preliminaries

We start with some preliminaries. These include relation reduction algorithm, row blocking data transfer mode and block merge join algorithm.

2.1 Relation Reduction Algorithm

Definition 1. Let R, S be the two relations that need to be joined based on the join attribute T . Let R' and S' be the subsets of R and S respectively consisting of tuples that satisfy the join condition. If $|R'|$ and $|S'|$ represent the sizes of R' and S' respectively, then the *selection rate* of relations R and S upon the join attribute T is $\rho_R = |R'|/|R|$, $\rho_S = |S'|/|S|$ respectively.

We take example of two relations R and S located respectively at some arbitrary nodes A and B in data grid and need to be joined according to some join attribute T . If the selection rates of R and S are small, i.e., only small number of tuples in R and S satisfy the join condition then we need to transfer only *these* tuples instead of all the tuples in R and S to the execution node where join operation will be performed.

The steps of relation reduction algorithm are as follows:

Step1. Get $T[R]$ and $T[S]$ from nodes A and B respectively, where $T[R]$ and $T[S]$ are the projection results of R and S according to join attribute T by using commonly known sort-based projection algorithm.

Step2. Transfer $T[R]$ from A to B and $T[S]$ from B to A . At node A , get R' which is the join result of $T[S]$ and R , similarly at node B , get S' which is the join result of $T[R]$ and S . R' and S' are separately the subset of R and S which satisfy the join condition.

Because of using sort-based projection algorithm, the tuples in R' and S' are either ranked in non-decreasing order or in non-increasing order on join attribute T . We assume the order is non-decreasing in this paper.

Lemma 1. (1) The join of two relations R and S is equal to the join of R' and S' where R' and S' are subsets of R and S respectively satisfying the join condition. That is, $R \triangleright \triangleleft S = R' \triangleright \triangleleft S'$.

(2) R' and S' are minimal, that is, there is no pair of relations $\langle R'', S'' \rangle$, such that $R'' \subset R'$, $S'' \subset S'$, and $R'' \triangleright \triangleleft S'' = R \triangleright \triangleleft S$.

Proof: (1) $R' \triangleright \triangleleft S' = (T[S] \triangleright \triangleleft R) \triangleright \triangleleft (T[R] \triangleright \triangleleft S) = (T[R] \triangleright \triangleleft R) \triangleright \triangleleft (T[S] \triangleright \triangleleft S) = R \triangleright \triangleleft S$

(2) Assume that there exist the other two relations R'' and S'' , $R'' \subset R'$, $S'' \subset S'$, such that $R'' \triangleright \triangleleft S'' = R \triangleright \triangleleft S$. There exists a tuple t , $t \in R'$ and $t \notin R''$. t does not participate the join operation $R'' \triangleright \triangleleft S''$, due to $R'' \triangleright \triangleleft S'' = R \triangleright \triangleleft S$, so t also participate the join operation $R \triangleright \triangleleft S$, it is obviously contradiction. Hence $R'' = R'$ and R' is the minimal. Similarly S' is minimal. \square

2.2 Row Blocking Data Transfer Mode

For transferring datasets from one grid node to others, we suggest row blocking data transfer technique in which tuples are shipped from one grid node to others in blocks.

In traditional approach, tuples are shipped one by one through the network. Therefore any short delay in the network would immediately stop the execution of the query at the receiving node because of a shortage of tuples to consume. But in row blocking, the receiving node would have a reservoir of tuples and thus can feed operator even if the tuples of next block is delayed. In this way, the approach compensates for delay in data arrival up to a certain extent.

2.3 Block Merge Join Algorithm

Suppose node E is the execution node in which join operation is to be performed over relations R' and S' . First, we need to set up two input buffers namely IB_R and IB_S for R' and S' respectively. Further we assume two memory spaces MR and MS allocated for R' and S' respectively to store received tuples from R' and S' .

Now, for each block of R' and S' , there are two phases to complete the join operation at node E : (1) *insertion phase*, each block is inserted into its corresponding memory location. (2) *match phase*, given a variable h_1 , for each block $R'[h_1]$, first we find every matched block from MS , secondly each tuple in block $R'[h_1]$ is probed with the tuples in every matched block and if join condition is satisfied, these two tuples are merged and the result is produced into output buffer. Similarly, for each block $S'[h_2]$, the same operation is repeated.

We explain the algorithm in detail as follows :

1. Set two variants PR and PS for MR and MS respectively, with initial values 1, set the two Boolean variants FR and FS for MR and MS respectively, with initial values *False*. FR and FS are used to control the changes in PR and PS .

2. When node E receives a block and inserts it into its corresponding memory, it records the minimum and maximum of join attribute T . That is, it records the values of T in the *first* tuple and the *last* tuple and builds a value-range for the block. In a similar way we get a value-range for each block in MR and MS .
3. Before merge join operation of two blocks, we can inspect the two value-ranges of the blocks. Once there is an overlap between newly inserted block $R'[h_1]$ and some block $S'[h_2]$ in MS , we set FS equal to *True*. If there is no overlap and FS is *False*, set $PS=PS+1$.
4. When a new inserted block $R'[h_1]$ in MR wants to probe with some block in MS , it is better to probe from $S'[PS]$. Because tuples in block $S'[PS-k](1 \leq k \leq PS-1)$ have been probed and cannot satisfy join condition. Similarly, when a new inserted block $S'[h_2]$ in MS wants to probe with some block in MR , it should probe from $R'[PR]$.
5. $S'[k](k < PS)$ denotes the blocks that have been inserted into MS before $S'[PS]$, $R'[m](PR < m)$ denotes the blocks which will be inserted into MR after $R'[PR]$. When PS plus one, any tuple in $S'[k]$ and $R'[m]$ shall not satisfy the join condition because all tuples in each block in non-decreasing order on T . Therefore, if node E needs to receive a new block of S' and there is out of memory in MS we can free memory space by deleting all blocks $S'[k](k < PS)$. The operation for MR is similar.

Block merge join algorithm produces join results in an efficient manner. It is symmetric and well to make it unnecessary to name inner relation and outer relation. In case of network delay, the algorithm merges tuples present in memory while waiting for new tuples to come. If there is memory shortage, all tuples which have been inserted before $R'[PR]$ or $S'[PS]$ can be deleted, therefore, this could free memory space for next coming blocks and avoid shipping blocks between memory and disk.

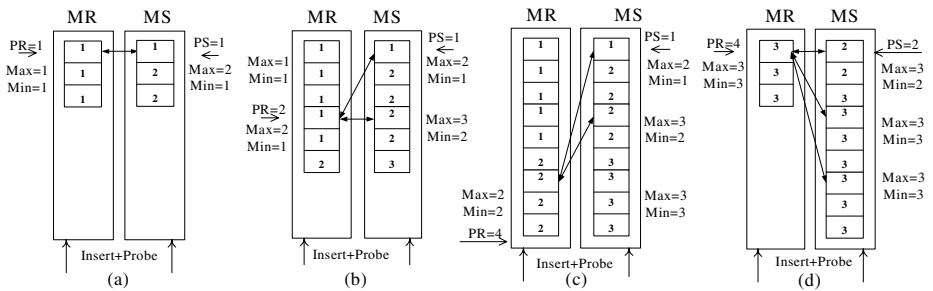


Fig. 1. An example of block merge join algorithm

Example. Assume relation R_i consists of four blocks and S_i consists of three blocks; MR and MS can contain three blocks at most and each block contains three tuples. Initially, MR and MS are both empty and $PS=PR=1$. The following figures explain the process of block merge join algorithm.

3 Edge-Weight-Minimum-Matching in Weighted Bigraph

In this section, we discuss Edge-weight-minimum-matching algorithms by employing some important concepts from Graph Theory, which can be used to solve the problem of replica matching and execution node selection in next section described.

Definition 2. A *maximal matching* in a graph is a matching which cannot be enlarged by adding an edge. It is a matching of maximum size among all matchings in the graph.

Definition 3. Given a weighted complete A, B -bigraph $G=(V, E), A=\{A_1, A_2, \dots, A_n\}, B=\{B_1, B_2, \dots, B_m\}$ where $n \leq m$. A matching M is called a *perfect matching from A to B*, such that all vertices in A are saturated by M .

Definition 4. Let matching M be a perfect matching from A to B . Let the weights on its edges compose a set $Set(W(M)), Set(W(M))=\{W(1), W(2), \dots, W(n)\}$. Assume $W(M_i)$ is the maximum among $W(1), W(2), \dots, W(n), W(M_i)=Max\{W(1), W(2), \dots, W(n)\}$, then $W(M_i)$ is called *edge-weight-maximum in M*.

Definition 5. Given a weighted complete A, B -bigraph $G=(V, E), A=\{A_1, A_2, \dots, A_n\}, B=\{B_1, B_2, \dots, B_m\}$ where $n \leq m$. There are N perfect matchings from A to B , where $N = C_m^n * n!$. These perfect matchings compose a set $Set(M), Set(M)=\{M_1, M_2, \dots, M_N\}$. All edge-weight-maximums in these N perfect matchings from A to B compose a set $Set(Max(W(M))), Set(Max(W(M)))=\{Max(W(M_1)), Max(W(M_2)), \dots, Max(W(M_N))\}$. Assume $Q=Min(Set(Max(W(M)))) = Max(W(M_i)), Q$ is minimum among all edge-weight-maximums and called *edge-weight-minimum in G*. The matching M_i is called *edge-weight-minimum-matching*.

Let us see the following example to illustrate the above concepts in detail. Fig. 2 shows a weighted complete R, S -bigraph $G=(V, E), R= \{R_1, R_2\}, S=\{S_1, S_2, S_3\}$. From this weighted complete bigraph, we can get six perfect matchings from R to S , $Set(M)=\{(R_1S_1=1, R_2S_2=8), (R_1S_1=1, R_2S_3=9), (R_1S_2=8, R_2S_1=6), (R_1S_2=8, R_2S_3=9), (R_1S_3=7, R_2S_1=6), (R_1S_3=7, R_2S_2=8)\}$, $Set(Max(W(M)))=\{8, 9, 8, 9, 7, 8\}$, $Q=Min(Set(Max(W(M))))=7$, edge-weight-minimum-matching is $(R_1S_3=7, R_2S_1=6)$ as shown in Fig 3.

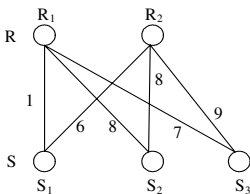


Fig. 2. A weighted complete bigraph G

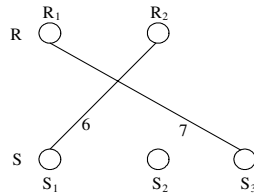


Fig. 3. The edge-weight-minimum-matching in G

By enumerating all the weighted complete matchings in the bigraph G and comparing all maximum weights of each matching, we can get the edge-weight-minimum in G and edge-weight minimum matching M . But when the number of matchings in-

creases, the problem becomes complicated. How to get the edge-weight-minimum matching M and the edge-weight-minimum in G is a problem worth being studied. Now let us introduce the algorithm for seeking an edge-weight-minimum matching M and an edge-weight-minimum in weighted complete bigraph G .

The main idea of the algorithm of Edge-Weight-Minimum Matching is: use Hungarian Algorithm or Ford-Fulkerson's Algorithm to find a maximum matching M which saturates R in a bipartite graph $G=(R, S)$ (G is not necessarily complete). If found, take away the edge E whose weight is maximum and get a subgraph $G_1=G-E$, then continue to find a maximum matching in $G_1 \dots$ until a maximum matching could not be found in subgraph of G .

Now we describe the algorithm as follows:

Input: a bipartite graph $G=(R, S)$ and the weights of G .

Output: an edge-weight-minimum matching M in G .

- ① Rank all the edges of the graph G as e_1, e_2, \dots, e_m such that the weights w_1, w_2, \dots, w_m are in nonincreasing order. Set k equal to zero and Flag equal to false
- ② Find a maximum matching M_k in G using Hungarian Algorithm or Ford-Fulkerson's Algorithm
 IF M_k is not found
 IF (Flag=false) stop, output "No solution!"
 ELSE stop, M_{k-1} is the solution
 ELSE
 IF (M_k saturate R) set Flag=true
- ③ Set $k:=k+1, G:=G-e_k$, go to Step 2.

Now let's analyze the time complexity of the algorithm. The time complexity of Hungarian Algorithm and Ford-Fulkerson's Algorithm are $O((n+m)^3)$ and $O(n \times m)$ respectively. So if Hungarian Algorithm is used, the time complexity is $O((n+m)^3 \times n \times m)$, and it is $O((n \times m)^2)$ if we use Ford-Fulkerson's Algorithm.

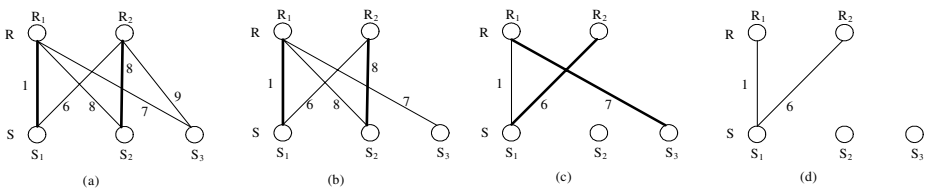


Fig. 4. Getting an edge-weight-minimum matching by using the first algorithm

In Fig.4(a) we have shown a weighted complete R, S -bigraph $G=(R, S), R=\{R_1, R_2\}, S=\{S_1, S_2, S_3\}$. The ranked result of weights on edges is $\{9, 8, 8, 7, 6, 1\}$. In Fig.4(a), $M_0=(R_1S_1=1, R_2S_2=8)$ can be found. Fig.4(b) shows a sub-graph $G=G-\{e_1=R_2S_3=9\}, M_1=(R_1S_1=1, R_2S_2=8)$ can also be found. Fig.4(c) shows a sub-graph $G=G-\{e_2=R_1S_2=8, e_3=R_2S_2=8\}, M_3=(R_1S_3=7, R_2S_1=6)$ can be found. Fig.4(d) shows a sub-graph $G=G-\{e_4=R_1S_3=7\}$, A matching M_4 in this sub-graph cannot be found, so the matching $M_3=(R_1S_3=7, R_2S_1=6)$ is the edge-weight-minimum-matching and $R_1S_3=7$ is the edge-weight-minimum in G .

4 Join Algorithm Using Multiple Replicas

We can describe the problem of join operation using multiple replicas in data grid as follows: assume there exist n full replicas R_1, R_2, \dots, R_n of R and m full replicas S_1, S_2, \dots, S_m of S in data grid, where $n \leq m$. A user at any arbitrary node C issues a query, which requires join of two relations R and S according to join attribute T , and gets join result at C .

Assume the full replicas $R_1, R_2, \dots, R_n, S_1, S_2, \dots, S_m$ can be found by using replica management services in data grid such as MDS in Globus. Also the consistency of replicas is maintained by replica management services. We take R_1, R_2, \dots, R_n as being consistent and having equal sizes. Let $|R_i|$ denote the size of R_i , then $|R_1|=|R_2|= \dots =|R_n|$. Similarly, S_1, S_2, \dots, S_m shall be consistent and equal, $|S_1|=|S_2|= \dots =|S_m|$.

The main idea of the proposed algorithm is as follows:

When we join the two relations R and S , first, we select n replicas from m full replicas of S so that each replica of S corresponds to a replica of R . Then by using relation reduction algorithm, these $2n$ replicas of R and S are reduced. Next, when execution nodes are selected by using matching theory, we ship every reduced relation pair of R and S to an execution node by using row blocking technique. While join operation is executed by using block merge join algorithm at the selected execution node, join results are transferred to (user) node C in pipeline mechanism.

Now we describe the phases of algorithm in detail.

4.1 Phase I: Matching for Replicas of R and S

Assume that the full replica R_i of R located at node A_i ($1 \leq i \leq n$) and the replica S_j of S located at node B_j ($1 \leq j \leq m$). In order to parallel the join operation of R and S , we select n replicas from m replicas of S before the join operation executes. Each selected replica of S corresponds to a replica of R .

The time taken to transfer data from node A to another node B is called *transfer time*. When data are transferred in parallel from nodes A_1, A_2, \dots, A_n to their n corresponding nodes among nodes B_1, B_2, \dots, B_m , the parallel transfer time could be calculated as the *maximum* of all transfer times. To reduce the total response time, the parallel transfer time should be as minimal as possible.

We can model this question as seeking an edge-weight-minimum-matching in weighted bigraph introduced in Section 3. Assume $A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m$ are $n+m$ nodes in a weighted complete A, B -bigraph $G=(V, E)$, $A=\{A_1, A_2, \dots, A_n\}$, $B=\{B_1, B_2, \dots, B_m\}$. The network transfer rate between nodes A_i and B_j is seen as the weight of edge A_iB_j considering the transference of data of same size. If an edge-weight-minimum-matching is found, then the edge-weight-minimum in G multiplying by the size of transferred data is the desired minimal parallel transfer time.

To get convenience in explanation, we assume that B_1, B_2, \dots, B_n are selected nodes and replica S_i located at node B_i corresponds to replica R_i located at node A_i .

4.2 Phase II: Parallel Hashing the Replicas of R and S

At nodes $A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_n$, we partition the relations R and S into n buckets by using the same hash function H . If necessary, the number of buckets is set to $n \times k$, where k is a natural number. For avoiding data skew, the size of each bucket is kept possibly same by selecting a suitable hash function.

Ultimately what we get: for each node A_i , the replica of R in it is partitioned into n buckets $R(1), R(2), \dots, R(n)$ and for each node B_i , the replica of S in it is partitioned into n buckets $S(1), S(2), \dots, S(n)$.

Note that if a tuple t_R in R and a tuple t_S in S satisfy the join condition, the values on join attribute of t_R and t_S must be equal. After partitioned by the same hash function, if the tuple t_R is partitioned into bucket $R(i)$, then the tuple t_S must be partitioned into bucket $S(i)$. Now it needs only to compare the tuples in buckets $R(i)$ and $S(i)$ when executing join operation on R and S .

4.3 Phase III: Relation Reduction Phase

At each node A_1, A_2, \dots, A_n , we parallel project the sub-relation $R(1), R(2), \dots, R(n)$ according to join attribute T by using sort-based project algorithm. For each replica R_1, R_2, \dots, R_n , the projections are all $T[R(1)], T[R(2)], \dots, T[R(n)]$. The intersection of $T[R(i)]$ and $T[R(j)](i \neq j)$ is empty. The union of $T[R(1)], T[R(2)], \dots, T[R(n)]$ equals $T[R_i]$, that is $T[R(1)] \cup T[R(2)] \cup \dots \cup T[R(n)] = T[R_1] = T[R_2] = \dots = T[R_n]$. Similarly, for nodes B_1, B_2, \dots, B_n , the projections are $T[S(1)], T[S(2)], \dots, T[S(n)]$. The intersection of $T[S(i)]$ and $T[S(j)](i \neq j)$ is empty and $T[S_1] = T[S_2] = \dots = T[S_n] = T[S(1)] \cup T[S(2)] \cup \dots \cup T[S(n)]$.

In Section 3, we have seen an edge-weight-minimum-matching M in which R_i located at node A_i is matched with S_i located at node B_i . So in order to reduce the sizes of relations R and S , $T[R_i]$ is transferred from node A_i to node B_i and $T[S_i]$ is transferred from node B_i to node A_i . After the join operation of $T[S_i]$ and R_i at node A_i , the reduced result of R_i is presented as $R'_i, R'_1 = R'_2 = \dots = R'_n = R(1) \cup R(2) \cup \dots \cup R(n)$. Similarly after the join operation of $T[R_i]$ and S_i at node B_i , the reduced result of S_i is obtained and presented as $S'_i, S'_1 = S'_2 = \dots = S'_n = S(1) \cup S(2) \cup \dots \cup S(n)$.

4.4 Phase IV: Execution Nodes Selection

Because data grid can provide coordinate resource sharing and cooperating computation, the resources such as database, CPU, disk, memory and instruments can be accessed and used by any user in data grid. Assume there are k ($n \leq k$) nodes that have tremendous processing capability and smaller network transfer rate, we can select n nodes from these k nodes as execution nodes. Reduced results of R and S are shipped in parallel to n execution nodes for the execution of block merge join operation.

Selecting n nodes from k nodes as execution nodes is the issue of concern. We can model this problem also as seeking an edge-weight-minimum-matching in weighted complete bigraph as mentioned in Section 3.

The network transfer rate between nodes A_i and E_j could be regarded as the weight of edge $A_i E_j$. Assume $A_1, A_2, \dots, A_n, E_1, E_2, \dots, E_k$ are $n+k$ nodes in a weighted com-

plete A, E -bigraph $G=(A, E)$, $A=\{A_1, A_2, \dots, A_n\}$, $E=\{E_1, E_2, \dots, E_k\}$. If an edge-weight-minimum-matching could be found in G , then the matching shows the node to which A_i corresponds. Suppose the nodes E_1, E_2, \dots, E_n are selected and A_i corresponds to E_h ($1 \leq h \leq n$).

When we select execution nodes for S_1, S_2, \dots, S_n , we can form an edge-weight-minimum-matching in $G=(B, E)$, $B=\{B_1, B_2, \dots, B_n\}$, $E=\{E_1, E_2, \dots, E_n\}$. We suppose B_j corresponds to E_h ($1 \leq h \leq n$).

4.5 Phase V: Parallel Block Merge Join Operation

For convenience of explaining the question, we assume the reduced results $R(i)$ and $S(i)$ are parallel shipped from nodes R_i and S_i to execution node E_i .

The reduced results $R(1), R(2), \dots, R(n), S(1), S(2), \dots, S(n)$ are shipped from nodes $A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_n$ to execution nodes E_1, E_2, \dots, E_n where block merge join operations are performed in parallel. The join results of $R(1)$ and $S(1), R(2)$ and $S(2), \dots, S(n)$ and $S(n)$ are shipped from E_1, E_2, \dots, E_n to (user) node C while block merge join operations are executing at nodes E_1, E_2, \dots, E_n .

5 Experiment and Results Analysis

5.1 Experimental Setup

We built a simulation environment and conducted an extensive performance study. All the experiments described in this paper are conducted under Windows 2000 environments with Pentium 2.4G CPU, 256 MB RAM and 40GB disk space.

5.2 Experiment I

In this experiment, the influence of network transfer rate to the proposed algorithm is mainly analyzed. Assume d_1 denotes the network transfer rate between node A_i having relation R_i and node B_i having relation S_i , d_2 denotes the network transfer rate between node A_i and execution node E_i , d_3 denotes the network transfer rate between node B_i and execution node E_i . For simplicity, we suppose $d_2=d_3$. Further, d_4 denotes the network transfer rate between node E_i and node C , and T_j denotes the time taken to transfer join result from node E_i to node C .

For each pair of buckets $R(i)$ and $S(i)$, the data size transferred between A_i and B_i is no more than $\max\{|T[R(i)]|, |T[S(i)]|\}$, the data sizes transferred between A_i and E_i, B_i and E_i are no more than $\max\{|R(i)|, |S(i)|\}$, the data size transferred between E_i and C is represented as $|R(i)S(i)|$.

Because $R(i)$ and $S(i)$ are the reduced results of $R(i)$ and $S(i)$ by using relation reduction algorithm, the number of different values on the join attribute T in $R(i)$ and $S(i)$ are equal. So we can get

$$\max\{|R(i)|, |S(i)|\} \leq |R(i)S(i)| \leq |R(i)| \times |S(i)| \quad (1)$$

In most situations, $\max\{|T[R(i)]|, |T[S(i)]|\}$ is much less than $\max\{|R(i)|, |S(i)|\}$. So we can get the following inequality:

$$\max\{|T[R(i)]|, |T[S(i)]|\} \leq \max\{|R(i)|, |S(i)|\} \leq |R(i)S(i)| \leq |R(i)| \times |S(i)| \tag{2}$$

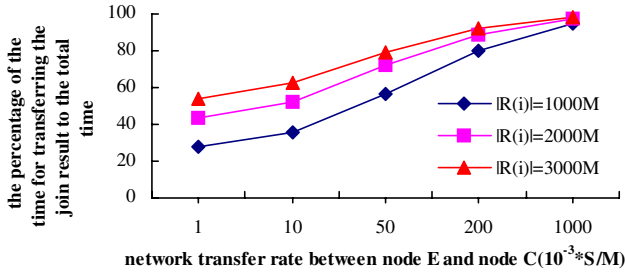


Fig. 5. The percentage of the time for transferring join result to the total response time

Assume the sizes of $R(i)$ and $S(i)$ are equal. Fig. 5 shows the percentage of T_j to total response time in the proposed algorithm. If the size of $R(i)$ and $S(i)$ is fixed, with the increase of d_4 , T_j will decrease and the percentage of T_j to total response time will decrease as well. In the proposed algorithm, we first select the nodes with much smaller network transfer rate between them and the node C as execution nodes. For selecting the locations of R and S , we select the nodes with smaller network transfer rate among them and the execution node E_i . Although the network transfer rate between A_i and B_i can influence the total response time to a certainty, the influence is much less than the influence of d_4 and d_2 .

5.3 Experiment II

We run this experiment using two methods: the first method represents not using relation reduction algorithm to reduce the sizes of relations, and the second method represents using relation reduction algorithm to reduce the sizes of relations. The influences to total response time between these two methods are compared. In the first method, relations R and S are not reduced and shipped to execution node E where join operation is completed and the results transferred to node C ; in the second method, relations R and S are first reduced. The reduced results R' and S' are shipped to execution node E where block merge join operation is performed and the result are transferred to node C .

Assume the selective rates ρ_R and ρ_S are same and are represented as ρ . From the experimental result, we can see that when ρ is much smaller, the time of transferring R' and S' and the total response time in the second method are much smaller. When ρ reaches a fixed value, the time of reducing relations becomes much more. The response time of the second method may exceed the response time of the first method. When $\rho=100\%$, that is $R=R'$ and $S=S'$, the time that the second method exceeds the first method is the time cost by using relation reduction algorithm.

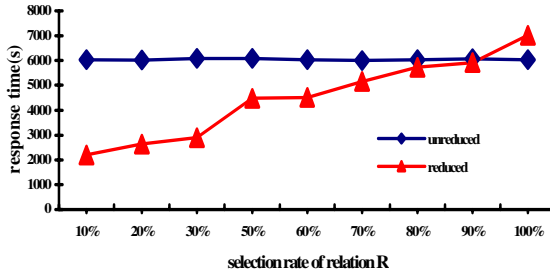


Fig. 6. The influence of selection rate to the total response time

6 Conclusion and Future Work

This paper has proposed a join algorithm using multiple replicas in data grid. This algorithm can solve the problem of replica matching and execution-node selection by making use of matching theory while keeping in view the heterogeneity of network bandwidth. To introduce parallelism in join process, we have developed and used reduction algorithm, row blocking technique and pipelined mechanism. The analytical and experimental results demonstrate the effectiveness of the proposed join algorithm for the management of data in data grid. There are a lot of problems unsolved in query processing in data grid. In future work, we will focus on the following subjects: (1) how to join the relations which are partial replicas, that is, they are composed of some parts in the original data, is a problem. (2) how to solve the order problem in multiple join operations in data grid. (3) the problem of caching query results, when a query result is produced, how to cache it so that next-coming query could get the results from the cache instead of querying data source again.

References

1. Foster, I., Kesselman, C.: *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1998
2. Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., Tuecke, S.: The Data Grid: towards an architecture for the distributed management and analysis of large scientific datasets. *Journal of Network and Computer Applications*, (23), 187-200, 2001
3. Hoschek, W., Jaen-Martinez, J., Samar, A., Stockinger, H., Stockinger, K.: Data Management in an International Data Grid Project. In 1st IEEE/ACM International Workshop on Grid Computing (Grid'2000), Bangalore, India, December 17-20 2000
4. Segal, B.: Grid Computing: The European Data Grid Project. IEEE Nuclear Science Symposium and Medical Imaging Conference, Lyon, 15-20 October 2000
5. Stockinger, H., Samar, A., Allcock, B., Foster, I., Holtman, K., Tierney, B.: File and Object Replication in Data Grids. *Journal of Cluster Computing*, Kluwer Academic Publishers, 5(3):305-314, 2002.
6. Vazhkudai, S., Tuecke, S., Foster, I.: Replica Selection in the Globus Data Grid. *Proceedings of the First IEEE/ACM International Conference on Cluster Computing and the Grid (CCGRID 2001)*, IEEE Computer Society Press, May 2001, pp. 106-113.

7. Lamahemedi, H., Szymanski, B., Zujun, S., Deelman, E.: Data Replication Strategies in Grid Environments, Proceeding 5th International Conference on Algorithms and Architecture for Parallel Processing, ICA3PP'2002, Beijing, China, October 2002, IEEE Computer Science Press, Los Alamitos, CA, 2002, pp. 378-383.
8. Ferdean, C., Makpangou, M.: A scalable replica selection strategy based on flexible contracts. The 3rd IEEE Workshop on Internet Applications(WIAPP), San Jose, CA, June 2003.
9. Smith, J., Gounaris, A., Watson, P., Paton, NW., Fernandes, AAA., Sakellariou, R.: Distributed Query Processing on the Grid, 3rd Int. Workshop on Grid Computing, J.Sterbenz, O.Takada, C.Tschudin, B.Plattner (eds.), Springer-Verlag, 279-290, 2002
10. Alpdemir, MN., Mukherjee, A., Paton, NW., Watson, P., Fernandes, AAA., Gounaris, A., Smith, J.: Service-based Distributed Querying on the Grid. In proceedings of the First International Conference on Service Oriented Computing, December 15-18, 2003, Trento, Italy, Springer, LNCS 2910 467-482.
11. Stockinger, H.: Distributed Database Management Systems and the Data Grid. In 18th IEEE Symposium on Mass Storage Systems and 9th NASA Goddard Conference on Mass Storage Systems and Technologies, San Diego, April 17-20 2001.
12. Ives, Z., Florescu, D., Friedman, M., Levy, A., Weld, DS.: An Adaptive Query Execution System for Data Integration. Proceedings of the 1999 ACM SIGMOD international conference on Management of data, Philadelphia, Pennsylvania, United States, 1999, pp. 299-310
13. Roussopoulos, N., Kang, H.: A pipeline n-way join algorithm based on the 2-way semijoin program. IEEE Transactions on Knowledge And Data Engineering, 3(4): 486-495, 1991.

A Heuristic and Distributed QoS Route Discovery Method for Mobile Ad Hoc Networks

Peng Fu and Deyun Zhang

Network Institute, School of Electronic and Information Engineering,
Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China
fupeng@xanet.edu.cn

Abstract. Quality of service (QoS) routing has been receiving increasingly intensive attention in mobile ad hoc networks (MANETs) fields. Many works have been done to overcome the passive factors due to the natural characters of MANETs such as performance constraints and dynamic network topology. In this paper, we present a new QoS route discovery algorithm for MANETs based on hybrid heuristic optimize algorithm. This method integrates the route discovery scheme with a swarm intelligent algorithm and the local search method based on SA. The ant algorithm has powerful ability of global search that will help to increase the probability of success in finding QoS feasible paths, and the route selection method based on probability can control flooding to reduce network overhead in the process of route discovery. The local search method based on SA is used to increase the convergence rate of route discovery algorithm and avoid the problem of stagnancy routes. The simulation experiment results based on NS2 show that network performance is improved obviously, and the method proposed is efficient and effective.

1 Introduction

Traditional host based computation paradigm cannot satisfy the current tremendous increment of application requirement. With the great development of computer technique, computation paradigm and data share paradigm are transiting to the stage of Ubiquitous Computing or Peer-to-peer Computing through the research on mobile computation and embeded computation. Mobile ad hoc networks (MANETs) is a kind of new technique of wireless communication [1][2], which provides a valuable and potential communication method for various heterogeneous mobile nodes in Ubiquitous Computing environment. In an ad hoc network, there are no any fixed infrastructures such as base stations or switching centers, and the network is self-configuring and self-organizing without any central management system [2], the mobile nodes act as the router and the host at the same time. Military, disaster relief area, commercial uses, and home network are the main applications of ad hoc networks due to their unique flexible properties that an ad hoc network can be deployed rapidly with relatively low cost.

Future MANETs will carry multimedia applications, so it is important that MANETs provide QoS support in order to provide quality delivery for various data [3]. There are many existing QoS technique, such as QoS model, resource reservation, and QoS route. The notion of QoS route [4] is to find, establish and maintain a route from a source to a

destination that have sufficient available resources can satisfies the end-to-end QoS requirement. The problem of QoS route in MANETs have received intensive attention, however, much difficulty is faced and the existing QoS routing algorithms in wireline network cannot be applied directly due to the natural characters of such network. Those characters include: (1) dynamic network topology for the mobile nodes; (2) dynamic join and exit of member nodes; (3) performance constrain for wireless link; (4) peer-to-peer and self-organizing network frame; (5) process capacity constrain of single mobile node; (6) power constrain. The route discovery method based on flooding route request packets will affect the performance of such network badly. It is proved that multiple QoS contains QoS routing problem is a NP-complete problem [8], and the heuristic algorithms is fit to solve such problem, but it is difficult to inquire the global network state information. Furthermore, it is important to get the convergence result rapidly for the dynamic characters. For these reason, a novel heuristic QoS route discovery method for MANETs, named SAANT, is proposed in this paper, which is based on a hybrid heuristic strategy of adaptive ant colony algorithm and simulated annealing. To cope with the nature characters of MANETs, our method is designed in order to include the following features: (1) ant colony algorithm is adaptive, flexible that is appropriate for the dynamic network environment [5] of changing topology and link state; (2) partial distributed routing algorithm that can reduce the utilization of the global network information; (3) rout table based on probability reducing the broadcast to reduce the effect on network performance in some degreed; (4) the simulated annealing algorithm is integrated to remedy the weak ability of local search of ant algorithm, which is benefit to get fast and efficient search result for changing of MANET topology. This paper is organized as following: related work is compared in section 2; the formal problem model is proposed in section3; in section4, the SAANT algorithm is discussed in details; Then a simulation experiments is reported to manifest the performance of our proposal. Finally, we include our main result and discuss extensions that can be made in future work.

2 Related Works

Currently, many study have been done about QoS routing for MANETs by some researchers such as MANET workgroup of IETF. The routing protocols for MANETs may be broadly classified as table driven protocols and on-demand driven protocols [6]. Furthermore, a majority of QoS routing algorithms were developed based on the two kinds of routing protocols. Existing studies [7] show that table-driven protocols are more liable to suffer performance degradation due to exchanging route information periodically. However, although the on-demand protocols are considered as that have lower overhead, but which still have higher route discovery latency and overhead due to flooding. Some meta-heuristic algorithms are also applied in this area [8][9], and the aim is to search the optimize results (feasible QoS path) with accepted overhead. Ant colony algorithm have the ability parallel global search, which been applied in network routing for MANETs [10][11] widely. These kind of routing methods have the low routing overhead, and the good adaptivity to dynamic network environment. But the local search ability of ant colony algorithm is weak which maybe lead to the slower convergence rate.

3 Problem Model of QoS Routing in MANETs

An ad hoc network can be illustrated as an indirection weighted graphic $G, G=\{V, E\}$ [4], where V is the set of mobile nodes, and E is the set of bi-direction links, for any link $e \in E$, have $e=(v_i, v_j)$, and the nodes $v_i \in V, v_j \in V, i \neq j$, v_i, v_j is neighbor nodes. In ad hoc network, the member element of set V, E is dynamic. If $s \in V$ is the source node, $p \in \{V - \{s\}\}$ is destination node of one communication, for $\forall e \in E$, define these metric function as following:

- Bandwidth function: $B(e): E \rightarrow R^+$. available bandwidth of link $e \in E$.
- Delay function: $D(e): E \rightarrow R^+$. delay of link $e \in E$.
- Cost function: $C(e): E \rightarrow R^+$. cost for packet to pass the link $e \in E$.
- Packet loss rate function: $PL(e): E \rightarrow R^+$. the packet loss rate of link $e \in E$.

The relationship of nodes and links and the link state is dynamic for the dynamic characters of MANETs. So, we give such hypothesizes: (1) nodes can discovery their neighbors via some specific scheme; (2) nodes can get the adjoined links stat real timely.

For a unicast link l , we have:

$$l=(s, p)=(v_i, v_{i+1}, \dots, v_k), s=v_i, p=v_k, v_i, v_{i+1}, \dots, v_k \in V, k > i. \tag{1}$$

The QoS constrains can be expressed as (2):

$$\left\{ \begin{array}{l} D(l) = \sum_{i=1}^{k-1} D(v_i, v_{i+1}) \\ C(l) = \sum_{i=1}^{k-1} C(v_i, v_{i+1}) \\ B(l) = \min(B(v_i, v_{i+1}), B(v_{i+1}, v_{i+2}), \dots, B(v_{k-1}, v_k)), \\ PL(l) = \prod_{i=1}^{k-1} PL(v_i, v_{i+1}) \end{array} \right. . \tag{2}$$

We give the definition of the problem of unicast QoS route discovery in MANETs:

Def 1: In an ad hoc network, s is the source node and the p is the destination node, define the Bandwidth function: $B(e): E \rightarrow R^+$; Delay function: $D(e): E \rightarrow R^+$; Cost function: $C(e): E \rightarrow R^+$; Packet loss rate function: $PL(e): E \rightarrow R^+$, and we define the problem of unicast QoS route discovery in MANETs as to find a feasible path l and satisfy the following condition:

$$(D(l) \leq d) \wedge (C(l) \leq c) \wedge (PL(l) \leq pl) \wedge (B(l) \geq b). \tag{3}$$

where, the constant d 、 c 、 pl 、 b express the QoS parameters of delay, cost, packet loss rate, and available bandwidth respectively.

It's a multi-constrain QoS routing problem, and belongs to NP complete prolem. In this paper, we will propose a hybrid heuristic strategy integrated ant colony algorithm and simulated annealing algorithm to solve the problem in dynamic network environment.

4 SA and Ant Algorithm Based QoS Routing for MANETs

4.1 Ant Algorithm Based QoS Route Discovery

Ant algorithm belongs to the swarm intelligent, which is proposed by Marco Dorigo first in his PHD thesis in 1992. This algorithm was inspired from the behavior of seeking food of ant colony. Real ants are capable of finding the shortest path from a food source to their nest by exploiting pheromone information and it is more likely for those following ants to select the paths with higher strength pheromone. The Ant algorithm has been applied in the pure routing problem widely. The artificial ants are realized as some especial packets that simulate the behavior of real ants to deposit and sensing the artificial pheromone. The route selection of ants is not based on the traditional route tables deployed at every routers but the transition probability that is gained rely on the strength of pheromone and the heuristic factors. According to the principle of ant colony algorithm [12][13] and existing works [10][11], we construct the new pheromone update equation, heuristic factors, and the transition probability function based on the problem model in Def 1. The transition probability function between v_i and v_j is:

$$P_{v_i v_j}^k(t) = \begin{cases} \frac{\tau_{v_i v_j}^\alpha(t) \eta_{v_i v_j}^\beta(t)}{\sum_{\mu \in V_{allow}} \tau_{v_i \mu}^\alpha(t) \eta_{v_i \mu}^\beta(t)}, & \mu \in V_{allow} \\ 0, & otherwise \end{cases} \quad (4)$$

where, $P_{v_i v_j}^k(t)$ express the probability of ant moving form node v_i to node v_j at time t ; $V_{allow} \subset V$ denotes the set of mobile nodes, at which the ant k is able to arrive for next hop. $\tau_{v_i v_j}^\alpha(t)$ denotes the pheromone quantity of path (v_i, v_j) at time t ; $\eta_{v_i v_j}^\beta(t)$ is a heuristic factor, α and β are two parameters that show the relative weightiness of pheromone and heuristic information during the process of QoS route selection.

Def 2: Imagine ant can arrive at the destination in n steps through the path l ; we define the rule of pheromone update at path l as:

$$\tau_{v_i v_j}(t+n) = \rho \tau_{v_i v_j}(t) + \Delta \tau_{v_i v_j}(t, t+n) \cdot \quad (5)$$

$$\Delta \tau_{v_i v_j}(t, t+n) = \sum_{k \in S} \Delta \tau_{v_i v_j}^k(t, t+n) \cdot \quad (6)$$

where, $\Delta\tau_{v_i,v_j}(t,t+n)$ denotes the pheromone increment on path (v_i,v_j) in this cycle; ρ is the attenuation(evaporation) factor, $0 < \rho < 1$, to avoid the infinite increment pheromone on path (v_i,v_j) that maybe lead to stagnation route potentially; $\Delta\tau_{v_i,v_j}^k(t,t+n)$ denotes the pheromone deployed be ant k on path (v_i,v_j) during the period $(t,t+n)$.

$$\Delta\tau_{v_i,v_j}^k(t,t+n) = \begin{cases} \frac{Q}{|l|} & \text{If ant } k \text{ passed the} \\ & \text{path } (v_i,v_j). \\ 0 & \text{Otherwise.} \end{cases} \tag{7}$$

where, Q is a constant; $|l|$ denotes the amount of nodes on the path l . $Q/|l|$ denotes the average pheromone quantity of unit length on path l .

The problem in definition 1 is different from the pure routing problem, since the QoS constrains is the more important factor than length or hop count in route selection. So we revised the formula (7) as:

$$\Delta\tau_{v_i,v_j}^k(t,t+n) = \begin{cases} \frac{(B(l))^{\sigma_B} + (d - D(l))^{\sigma_D} + (1 - PL(l))^{\sigma_{PL}}}{|l|(C(l))^{\sigma_C}}, & \text{If ant } k \text{ passed} \\ & \text{the path } (v_i,v_j). \\ 0 & \text{Otherwise.} \end{cases} \tag{8}$$

where, $\sigma_B, \sigma_D, \sigma_{PL}, \sigma_C$ denotes the are weight factors that show the relative weightiness of each QoS parameters during pheromone update.

Also, we revised the formula (4) as:

$$\eta_{v_i,v_j}^\beta(t) = \left(\frac{(B(v_i,v_j))^{\beta_B} + (d - D(v_i,v_j))^{\beta_D} + (1 - PL(v_i,v_j))^{\beta_{PL}}}{(C(v_i,v_j))^{\beta_C}} \right)^\beta. \tag{9}$$

where, $\beta_B, \beta_D, \beta_{PL}, \beta_C$ denotes the are weight factors that show the relative weightiness of each QoS parameters during route selection.

Thus, we get the transition probability function (4) and (9), and the relevant rules of pheromone update (5)-(8).

It is pivotal for improving the performance of ant algorithm to integrate the pheromone update and heuristic factors. In this paper, we design the rules of pheromone update that is based on two reasons: (1) Our rules of pheromone update found on the ant-cycle method [13], which has been proved to be more efficient than those methods based on local state information such as ant-density system. (2) Utilizing the global state information of a link is benefit for avoiding the problem of premature convergence and local optimize path in some degree. However, in such a dynamic network environment, it's very difficult to acquire global network state in time. Therefore, we construct the heuristic factors based on local state for transition probability function.

Thus, the transition probability function utilizes two aspects of information, and the ants would be able to process route selection only based local information if the global have not been acquired in time.

4.2 Principle of Simulated Annealing Algorithm

Simulated annealing (SA) algorithm that can be regarded as a kind of most famous heuristic reformative local search algorithm was first introduced by Metropolis et al. in 1953, and then be used to solve optimization problems by Kirkpatrick in 1983 firstly. The concept of that is based on the behavior in which metal cools and freezes into a minimum energy crystalline structure (thermodynamic equilibrium) in the process of annealing. Hence the process can be thought of as an adiabatic approach to the lowest energy state [14]. The overview of the simulated annealing algorithm scheme is as following: (1) give energy equation to estimate solutions; (2) select an initial solution as the current solution and give the initial temperature; (3) generate a new solution in the neighborhood of current solution in certain way; (4) estimate the new solution with equation, the better solution will be accepted, but the worse solution will be accepted in some probability; (5) adjust temperature, and continue the cycle till the terminate condition is satisfied.

Simulated annealing algorithm's major advantage over other methods is an ability to avoid the traps at local optimize. SA algorithm can accept the deterioration solution in some degree; at $T=0$, the global optimize solution will be get. The SA has the powerful ability of local search, which will be integrated our method to improve the probability of discovering QoS route for MANETs.

4.3 SA and Ant Colony Algorithm Based QoS Route Algorithm for MANETs

1) The Principle of Algorithm

Ant colony algorithm has the ability of parallel searching solution space and can avoid the problem of local optimize. However, exiting study shows that ant colony algorithm has some disadvantages: (1) relative weak ability of local search and slow convergence rate; (2) stagnancy route [5], that is almost all ants select one optimal path and this recursively increases an ant's preference for this path. This may lead to congestion, load unbalance, and stagnation of exploring the solution space exploration. All of that may be against the dynamic network environment of MANETs. To overcome these problems, some local search method was integrated in ant colony. For example, in the paper [15], the immune computing is intergraded to solve the problem of weapon target assignment (WTA). SA algorithm has the powerful ability of search solutions in relative small search space rapidly and avoid the problem of local optimize. But the SA is lack of inquiring the global solution space, which is not good for the searching process enter the required region. Therefore, we introduce SA to the ant colony algorithm based QoS route discovery method that has been discussed in section 4.1, and propose a hybrid optimize strategy, SAANT, that integrate SA based local search method to ant colony algorithm. The aim is to adjust the intermediate solution in the parallel search process of ants with SA local search that will help to increase the convergence rate of route algorithm and the probability of discovery QoS routes, which is more fit for the nature of MANETs.

The algorithm of SAANT is composed of two main parts: ant colony algorithm based QoS route discovery algorithm and SA based local search algorithm. The ant colony algorithm runs at each distributed mobile node in an ad hoc network, when the source node initiates a QoS routing request, the ants will be sent as probe packets to find the feasible path; The SA based local search algorithm only runs at source node, and the intermediate solution of ant colony algorithm will be used as the initiate solution of SA. SAANT utilizes the adjustment of SA based local search to direct and expedite the ant colony algorithm based global search.

(2) Data Assumptions

To describe the SAANT algorithm clearly, it's necessary to denote the main data object precisely as following.

We give the definition of energy equation used in SA first.

Def 3: Define the **energy equation** of a path l in an ad hoc network as:

$$f(l) = \max\left(\frac{D(l)}{C_d}, \frac{C(l)}{C_c}, \frac{B(l)}{C_b}, \frac{PL(l)}{C_{pl}}\right). \tag{10}$$

where, C_d, C_c, C_b, C_{pl} denotes the are weight factors of each QoS parameters during route selection.

Def 4: Pheromone table (Ptable), for a node v_i in MANET must maintain relative pheromone table $Ptable_i$; for each adjoined path $e_{i\mu}, \mu \in V_{allow}$ to node v_i , the $Ptable_i$ stores the realtime pheromone value $\tau_{v_i\mu}$.

Def 5: Pheromone route table (RPtable), each node v_i needs to maintain relative pheromone route table $RPtable_i$; for each adjoined path $e_{i\mu}, \mu \in V_{allow}$ of node v_i , the $RPtable_i$ stores the realtime transition probability $P_{v_i\mu}^k(t)$, and $P_{v_i\mu}^k(t)$ is computed according to formula (4).

Def 6: Ant state vector (Asvector), Asvector is the data structure of each ant, for ant k , there exits:

$$Asvector_k(antID, sourceID, destinationID, visitNodes, pheromoneSUM, TTL)$$

each parameter denotes following data object respectively: ID of ant, ID of source node, ID of destination node, the queue of the nodes that has been visit by this ant, the queue of relative link's pheromone value, time to life of ants.

Def 7: Feasible path stack (Fpstack), each node v_i needs to maintain it's feasible path stack $FPstack_i$, and the $FPstack_i$ begin to work only when the node v_i act as the source node. $FPstack_i$ is a kind of stack structure, which store all the feasible paths $Path_{v_i}^k$ found in one ant cycle.

(3) Description of SAANT Algorithm

SAANT include a main algorithm and a sub-algorithm for main algorithm to invoke.

Algorithm: SAANT

Input: QoS routing request vector, $Reuest(sourceID, destinationID, D, C, B, PL)$.

Output: final $Path_{sourceID}^k$

Initialize: Initiate time: $t = 0$.

Set timeout: T .

Initiate the Ptable of each node: $\tau_{v_i v_j}^\alpha(t) = C$, C is initiate value.

Initiate the pheromone update: $\Delta\tau_{v_i v_j} = 0$

Step1: When node v_S initiate a QoS routing request, the node v_S will send m ants periodically. But that is not ordinary broadcast, and the ants will be forwarded according to relative transition probability in $RPtable_{v_S}$. For ant $k(k \leq m)$, the initial- $k.TTL = 0$, if $k.TTL > T$ during the period of algorithm executing, ant k will be killed.

Step2: When ant $k(k \leq m)$ arrive at a node $v'(v' \in V)$, we have:

if $v' \neq destinationID$

if $v' \in k.visistNode s$

delete the records following the first occurring v' , and re-forward ant k according to the transition probability in $RPtable_{v'}$.

else forward ant k according to the transition probability in $RPtable_{v'}$.

else goto step3

Step3: if $v' = destination nID$

compute the revise path l'_k from the forward path l_k stored in $k.visistNode s$.

$a = k.destination nID; k.sourceID = a; k.destination nID = k.sourceID$;

send ant k back to source v_S along the path l'_k .

Step4: When source node have received ant k :

① All the Ptables of the nodes along the path l_k will be updated based on the formula (5)(6)(7)(8), and each PRtable will be updated with formula (4) as well as.

② Push the path l_k in to stack $FPstack_{v_S}$, and the path with the largest pheromone value will be set at the top of stack.

Step5: When source node have received all the ants in one cycle, or the cycle is timeout ($t = nT$)

① If there exit paths in stack $FPstack_{v_S}$ that satisfy the QoS requirement, rename the path with largest pheromone value as $qmPath$, and go to step6;
Otherwise, invoke the algorithm SALS.

② If the return result of SALS satisfy the QoS requirement, rename the path with largest pheromone value as $qmPath$, and go to step6;

Otherwise, use the average largest pheromone value to update the path return by SALS.

go to Step1.

Step6: Return qmPath.

End

Algorithm: SALS

Input: $FPstack_{v_s}$, the intermediate search results of ant colony algorithm.

Output: the path that is the result of local search based on SA algorithm.

Step1: Let Ω be the set of feasible path, $f(x)$ be energy equation. Let initial temperature be ST_0 , and initiate solution be the stack head element of $FPstack_{v_s}$.

Step2: The following operations will be executed at the temperature ST_k :

Utilize genetic algorithm (GA) to carry out exploring in the neighborhood $N(x) \subset \Omega$ to produce the new solution x' .

(1) Let the feasible paths stored in $FPstack_{v_s}$ be the initiate population $P(g)$ with $G(G \leq m)$ individuals.

(2) Produce the new generation population $P(g + 1)$:

① The energy equation $f(x)$ will be used as fitness function.

② The crossover operation is to select two individuals (path) randomly, suppose two paths (nodes sequences) x_i and x_j share some intermediate node, so that for some node v , we have $x_i = \mu_1 v \mu_2$ and $x_j = \mu_3 v \mu_4$. The crossover operation will then generate the new path $x'_i = \mu_3 v \mu_2$ and $x'_j = \mu_1 v \mu_4$. And get the new fitness $f(x_i)$, $f(x_j)$. Compute the difference value of energy equation: $\Delta f = f(x') - f(x)$; if $\Delta f < 0$, accept the new solution, otherwise, accept the new solution [16] with the probability $\min\{1, \exp(-\Delta f / ST_k)\} > Random[0,1]$, the $Random[0,1]$ is a random real in $[0,1]$.

③ In the mutation operation, the genes are chosen randomly in the range from zero up to mutation probability $P_{mutation} \leq 1/\ell$, where ℓ is the path (individuals) length. The accept rules for new solution is same as ②.

Step3: Decrease the temperature with some method, such as $ST_{k+1} = \rho ST_k, \rho \in [0,1]$.

Step4: if $ST_k = 0$, terminate search and return the current solution.

End

In the process of simulated annealing, if the change in energy is positive it is accepted with a probability given by the Boltzmann factor $\exp(-\Delta f / ST_k)$. This process is then repeated sufficient times to give good sampling statistics and the accept probability will keep to the Boltzmann probability distribution:

$$\Pr(x) = \frac{1}{Z(ST)} \exp\left(-\frac{f(x)}{k_B ST}\right). \quad (11)$$

where, $Z(ST) = \sum_{x \in \Omega} \exp\left\{-\frac{f(x)}{k_B ST}\right\}$ is the regularization coefficient, and $k_B > 0$ is the Boltzmann coefficient.

5 Simulation and Analysis

In this section, we use a simulation-based method to study the relative performance of the proposed schemes. We have developed a simulated model based on NS2(network simulator 2) [17] simulator. In the following simulations, the simulation environment configurations are showed in table 1.

Table 1. Simulation environment configuration

Simulation environment	Option and parameter
Flat Size	500 x500 meters.
Number of Nodes	100
MAC Layer	IEEE 802.11, peer to peer mode
Transport layer	User Datagram Protocol (UDP)
Traffic	Traffic enters the simulated network from 10 nodes and flows symmetrically from the 10 nodes to the other 10 nodes with the packet size of 512 bytes.
Mobility model	Waypoint
Velocity	Between 0 and 0.7 m/s
Simulation Time	500 sec

The object of simulation is to compare the SAANT with the ant colony algorithm only, and the complete SAANT algorithm with both ant colony algorithm and the SA based local search, and the traditional on-demand route discovery method (such as AODV) where the route discovery is based on the route request packet that is broadcasted completely.

Simulations are run for the three algorithms. The following metrics are used to estimate algorithm performances: (1) Average Delivery Delay: the average delay of all packets delivered to destinations; (2) Data Delivery Rate: Percentage of data packets delivered to the destinations. The Fig. 1 and Fig. 2 showed the simulation results we have obtained.

Fig. 1 shows the comparison of delivery delay for AODV, SAANT without SA and SAANT at difference interval time. All the three policy have a large initial delay for routes information is founded. SAANT has lower delay than both the other two algorithms, and AODV often shows larger delays. Fig. 2 shows the comparison of data delivery ratio for AODV, SAANT without SA and SAANT in various mobile velocities. As mobility increases the data delivery ratio of all the methods will drop down.

That means that the required QoS is not guaranteed due path breaking or nodes move to other position. However, the delivery ratio of SAANT drops down more slowly, which means that SAANT have better performance. In the case of MANETs, it is more important to find a route faster with accepted cost than to search the optimal route with unaccepted time for the dynamic network topology. Therefore, the simulation results show that SAANT is more efficient and effective than the traditional ant colony algorithm or on-demand routing method in processing problem of QoS routing for MANETs.

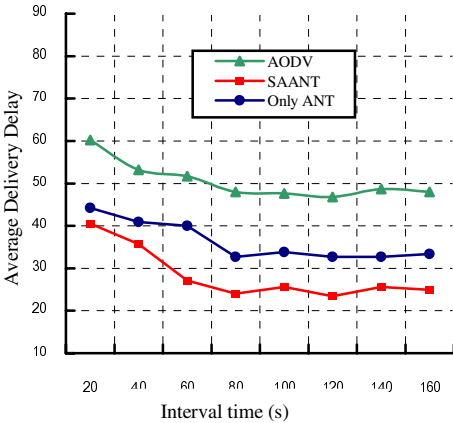


Fig. 1. Average Delivery Delay

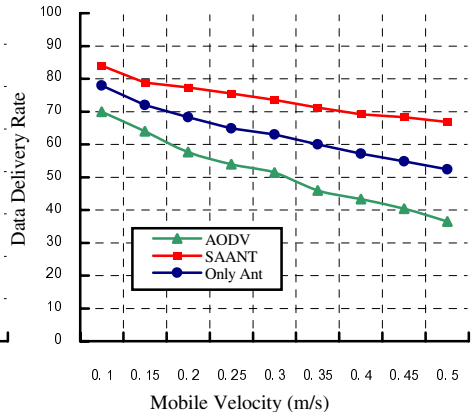


Fig. 2. Data Delivery Rate

6 Conclusions

MANETs will be a kind of pivotal communication technique for future pervasive applications. Thus, the QoS issues in MANETs started to receive increasing cattention in the literature. Any QoS routing algorithm in MANETs should be fit to the natural characters of MANETs. The dynamic network environment, relative low performance, and mesh network topology etc. make the solution more difficult than that in wireline network. In this paper, we apply a hybrid heuristic optimize algorithm based on ant colony algorithm and SA to solve the problem of QoS route discovery in MANET. Ant algorithm has powerful ability of parallel global search that will help to increase the probability of success in finding QoS feasible paths, and the route selection method based on probability can control flooding to reduce network overhead in the process of route discovery. The local search method based on SA is introduced to increase convergence rate of route discovery algorithm and avoid the problem of stagnancy routes. Then, we manifested the performance of proposed algorithm via simulation experiment based on NS2. Those problems, such as load balance, traffic control, resource reservation, and QoS violation recovery, will be taken into account in our further coming research.

References

1. IETF.: Mobile ad hoc networks charter, <http://www.ietf.org/html.charters/manet-charter.html>. (2004)
2. David R., Ignas G. N.: Ad hoc networking in future wireless communications. *Computer Communications*. 26(2003)36-40
3. Chakrabarti, S., Mishra, A.: QoS issues in ad hoc wireless networks. *IEEE Communications Magazine*. 39(2)(2001)142-148.
4. Chenxi, Z., Corson, M.S.: QoS routing for mobile ad hoc networks. *IEEE INFOCOM 2002*. 2(2002)958-967.
5. Kwang, M., S., Weng, H., S.: Ant colony optimization for routing and load-balancing: survey and new directions. *IEEE Transactions on Systems, Man and Cybernetics, Part A*. 33(5) (2003)560-572.
6. E.M. Royer, C.K.: A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. *IEEE Personal Communications*. 4(1999)46-55.
7. A. Boukerche: Simulation based comparative study of ad hoc routing protocols, *Proceedings of the 34th Annual Simulation Symposium*, Seattle, WA. April(2001)85-92.
8. Barolli, L., Koyama, A., Shiratori, N.: A QoS routing method for ad-hoc networks based on genetic algorithm. *14th International Workshop on Database and Expert Systems Applications*, 2003. *Proceedings*. (2003)175-179.
9. Usaha, W., Barria, J.: A reinforcement learning ticket-based probing path discovery scheme for MANETs. *Ad Hoc Networks*. 2(2004)319-334.
10. Hussein, O., Saadawi, T.: Ant routing algorithm for mobile ad-hoc networks (ARAMA). *2003 IEEE International Performance, Computing, and Communications Conference*. (2003) 281-290.
11. Shen, C., C., Jaikao C.: Ad hoc Multicast Routing Algorithm with Swarm Intelligence. *ACM Mobile Networks and Applications (MONET) Journal*. 10(2005)47-59.
12. Dorigo M., DiCaro, G., Gambardella, L., M.: Ant Algorithms for Discrete Optimization. *Artificial Life*. 5(2) (1999)137-172.
13. Li, S., Y., Chen, Y. Q., Li, Y.: *Ant Colony Algorithms with applications*. Harbin Institute of Technology Press. 2004.
14. Wang, L.: *Intelligent Optimization Algorithm with Applications*. Tsinghua University Press & Springer Press. 2001.
15. Lee, Z., J., Lee, C., Y., Su S. F.: An Immunity-based and Colony Optimization Algorithm for Solving Weapon-target Assignment Problem. *Applied Soft Computing*. 2(2002)39-47.
16. Wu, Z., Y., Shao, H., H.: Genetic Annealing Evolutionary Algorithm. *Journal of Shanghai Jiaotong University*. 2(1997)69-71.
17. The Network Simulator – NS2. <http://www.isi.edu/nsnam/ns/>.

Planning Enhanced Grid Workflow Management System Based on Agent¹

Lei Cao, Minglu Li, Jian Cao, and Ying Li

Department of Computer Science and Engineering, Shanghai Jiao Tong University,
Shanghai 20030, China
{lcao, li-ml, cao-jian, liying}@cs.sjtu.edu.cn

Abstract. Grid computing is becoming a mainstream technology for large-scale distributed resource sharing and system integration. Workflow management is emerging as one of the most important grid services. In this paper, we present a planning enhanced grid workflow management system based on agent (AGWMS). AGWMS has a four-layer architecture. The agent layer makes the system more robust, flexible and intelligent. Artificial intelligence (AI) planning technology is utilized in the system to build the agent plans automatically. The adapter layer makes it easy that AGWMS invokes external applications. Our planning enhanced AGWMS is a novel one.

1 Introduction

Grid computing [1] facilitates the sharing and aggregation of heterogeneous and distributed resources, such as computing resources, data sources, instruments and application services. With the advent of grid technologies, scientists and engineers are building more and more complex applications to manage and process large data sets, and to execute scientific experiments on distributed grid resources. Means of composing and executing distributed applications to form complex workflow are needed. Some efforts in the research of business workflow and web services orchestration can be reused in grid workflow systems. Grid workflow is “a workflow intended to solve sophisticated scientific problems that occur in highly heterogeneous, distributed, complex, and dynamic Grid environments that comprise of one or more virtual organizations (VOs)” [2]. It is clear that scientists conduct a sequence of grid complex activities in VOs using various VO resources located at different sites in different administrative domains. We develop an adequate grid workflow management system for VOs to help scientists streamline, manage, and monitor their routinely scientific problem solving processes without having to know any details of the underlying complex structure and dynamic state of VOs.

¹ This paper has been supported by the 973 project (No.2002CB312002) of China, grand project of the Science and Technology Commission of Shanghai Municipality (No.03dz15027). It is also partly supported by "SEC E-Institute: Shanghai High Institutions Grid", Chinese High Technology Development Plan (No.2004AA104340), Chinese Semantic Grid Project (2003CB317005) and Chinese NSF Project (No.60473092).

However, there are new challenges needing to be addressed in a typical grid environment: (1) Computational and networking capabilities can vary significantly over time; (2) There is no central ownership and control in grids; (3) The execution of a grid workflow faces many uncertain factors such as unavailability, incomplete information and local policy changes, so a full-ahead plan [3] is not always suitable; (4) The processing of resource discovery and selection could be quite complicated in grids. Workflow and agent technologies are complementary to each other, and there has been a lot of work on integrating the two. In our grid workflow management system, we use multi-agent technology to meet those challenges. Our AGWMS has a multi-layer architecture to make itself more adaptive. The four-layer architecture is presented in Section 3. The system is distributed improving its execution efficiency. The workflow engines of AGWMS may scatter in many hosts to balance the system's workload using JINI [4] technology. Our AGWMS also applies AI planning technology to enhance its intelligence and robustness. Planning has been used to generate the agent plan automatically.

The paper is organized as follows. We discuss related work in Section 2. Section 3 presents the architecture of AGWMS in detail. Section 4 gives some planning definitions and algorithms. Section 5 presents AGWMS's multi-agent platform. A case study and the implementation of system are given in Section 6. We conclude in Section 7 with lessons learned and future research plans.

2 Related Work

DAGMan [5] is developed to schedule jobs to Condor system in an order represented by a DAG and to process them. With the integration of Chimera [6], Pegasus [7] maps and executes complex workflow based on full-ahead planning. In Pegasus, a workflow can be generated from metadata description of the desired data product using AI-based planning technologies. The Taverna project [8] has developed a tool for the composition and enactment of bioinformatics workflow for the life science community. The tool provides a graphical user interface for the composition of workflows. The workflow management system for grid computing, called GridFlow [9], is presented, including services of both global grid workflow management and local grid sub-workflow scheduling. Simulation, execution and monitoring functionalities are provided at the global grid level. McRunjob [10] is a grid workflow manager used to manage the generation of large numbers of production processing jobs in High Energy Physics. It converts core metadata into jobs submitted in a variety of environments. GALE [11] is an HPC workflow vocabulary that uses key grid services to provide a "run code X anywhere, then post process results, then ..." grid-level scripting language for users and problem solving environments. In addition, many efforts on the composition of Web services [12] can also be complementary to the development of grid workflow management systems.

3 The Architecture of AGWMS

From the Fig. 1, you can see that there are four layers in AGWMS.

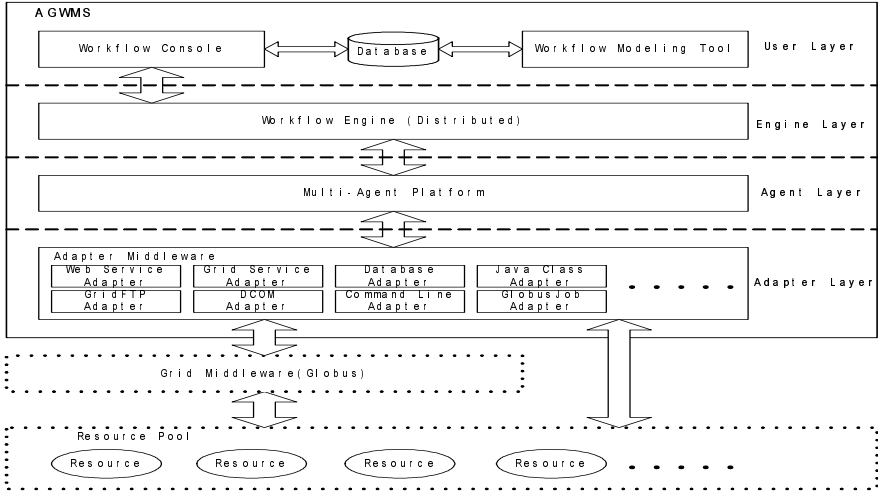


Fig. 1. Four-layer architecture of AGWMS

3.1 User Layer

The user layer is the interface between user and system. There are mainly two modules in this layer: Workflow modeling tool and Workflow console. *Workflow Modeling Tool* is used to model the complex grid application, which can be seen as a collection of activities (mostly computational tasks) that are processed in some order. It's based on Event-Condition-Action (ECA) rules. By modeling, related data are stored in files or database with multiple formats. In the system, we have two kinds of modeling tool. One is macro-workflow modeling tool, which is used to model abstract workflow of some application domains. The other is micro-workflow modeling tool, which is used to model the plan of domain-specific agent. Users can submit the preferred workflow model to the system for execution using *Workflow Console*. It's also used as the monitoring tool about execution states of the submitted workflow. We have two types of it: Web-based and Application-based.

3.2 Engine Layer

The engine layer is the core layer to manage workflow execution. Our workflow engine is distributed using JINI technology. It's also based on ECA rules. The engine is responsible for the instantiating and scheduling of the workflow, and it needs to monitor all activities' execution. If needed, the engine will interface with other external applications or operators when executing the workflow. We can balance the workload of the engine using JINI technology. It's adaptive and efficient when there are many coinstantaneous execution requests coming.

3.3 Agent Layer

The agent layer makes it possible that the grid workflow could be divided into two types: abstract one and concrete one. The abstract workflow in specific domain may

be stable and unchangeable, and also be called macro-workflow. It may not bind to actual grid resources. All activities in the abstract workflow are encapsulated into agents. There are many domain-specific agents in the multi-agent platform. The agent has its own micro-workflow, which is also called “agent plan”. We can predefine it using micro-workflow modeling tool. AI planning technology has also been used to create the plan automatically at run time. Section 5 will give the details.

3.4 Adapter Layer

The adapter layer makes it easy that AGWMS invokes external applications. Adapter is the proxy for the invocable application. It encapsulates the complex invoking details. Upper functionality modules only need to know the interface parameters of the adapter. Doing so makes our system more scalable. An adapter may have multiple input and output ports, through which streams of data tokens flow. Our adapter library allows one to plug different execution models into workflows as reusable adapters such as Web Service, Grid service, Database operation, DCOM, Java class, Command line, GridFTP, GlobusJob [13] and so on. More application types only need more adapters plugged in.

4 Planning Definitions and Algorithms

4.1 Definitions

We adopt partial-order planning (POP) algorithm in plan agent (PA). We give some useful definitions as following [14,15]:

Definition 1: A **planning problem** can be described as a three-tuple: $\langle \Gamma, \Delta, \delta \rangle$, where Γ denotes the initial state of the planner, Δ denotes the set of goal states the planning system should attempt to reach, δ is the set of (ground) actions the planner can perform in attempting to reach a goal state.

Definition 2: A **state** can be represented as a conjunction of positive literals. For example, $At(Nanjing, Frank) \wedge Destination(Sanya)$ might represent a state in the travel planning problem.

Definition 3: A **goal** is a partially specified state, represented as a conjunction of positive ground literals such as $At(Sanya, Frank) \wedge Hotel(PearlHotel)$.

Definition 4: An **action** is specified in terms of the preconditions that must hold before it can be executed, and the effects that ensue when it executes.

In our system, each plan has following four components, where the first two define the steps of the plan and the last two serve to determine how plans can be extended:

- A set of **actions**, which make up the steps of the plan. The empty plan contains just the *Start* and *Finish* actions. *Start* has no preconditions and has all the literals in the initial state of the planning problem. *Finish* has no effects and has the goal literals of the planning problem as its precondition.
- A set of **ordering constraints**. Each ordering constraint is of the form $A \blacktriangleright B$, which means that action A must be executed before action B .

- A set of **causal links**. A causal link between two actions A and B in the plan is written as $A \xrightarrow{p} B$ which asserts that p is an effect of the A and a precondition of B . The plan may not be extended by adding a new action C that conflicts with the causal link.
- A set of **open preconditions**. A precondition is open if it is not achieved by some action in the plan.

4.2 Algorithms

Algorithm 1 is used to generate the agent plan. We have modified POP successor function to permit the decomposition method used in the existing abstract workflow, and we can get the final concrete workflow by *Algorithm 2*.

Algorithm 1:

- 1) The initial plan contains *Start* and *Finish*, the ordering constraint $Start \prec Finish$, no casual links, and all the preconditions in *Finish* are open preconditions.
- 2) The successor function arbitrarily picks one open precondition p on an action B , and generates a successor plan for ever possible consistent way of choosing an action A that achieve p . Consistency is enforced as follows:
 - The causal link $A \xrightarrow{p} B$ and the ordering constrain $A \prec B$ are added to the plan. Action A may be an existing action in the plan or a new one. If it is new, add it to the plan and also add $Start \prec A$ and $A \prec Finish$.
 - We resolve conflicts between the new causal link and all existing actions, and between the action A (if it is new) and all existing causal links. A conflict between $A \xrightarrow{p} B$ and C is resolved by making C occur at some time outside the protection interval, either by adding $B \prec C$ or $C \prec A$. We add successor states for either or both if they result in consistent plan.
- 3) The goal test checks that a plan is a solution to the original planning problem. Because only consistent plans are generated, the goal test just needs to check that there are no open preconditions.

Algorithm 2:

- 1) Select one PA c' in Aw (the abstract workflow generated by user) and for the most suitable *Decomposition* (c, d) method from the plan library such that p and c' unify with substitution θ , we replace c' with $d' = SUBST(\theta, d)$.
 - First, the PA c' is removed from Aw. Then, for each step s in the decomposition d , we need to choose an action to fill the role of s and add it to the workflow.
 - Hook up the ordering constraints for d in the original workflow to the steps in d' .
 - Hook up causal links. If $B \xrightarrow{p} c'$ was a causal link in the original workflow, replace it by a set of causal links from B to all the steps in d with precondition p that were supplied by the *Start* step in the decomposition d .
- 2) Continue step 1) until all PAs in the original abstract workflow have been dealt with. Thus we get the last concrete workflow.

5 Multi-agent Platform Used in AGWMS

Currently, there is not much consensus on what an "agent" is, and many definitions abound. We define an agent as "a software component that acts autonomously on

behalf of a person or organization, and is also able to interact with its environment and with other agents". Agents here should be complex including features: autonomous, communication, self-consistent, goal-oriented, and reacting to environment. Other high-level features such as learning, negotiation are also of benefit. All agents in the agent layer are constructed according to Belief-Desire-Intention (BDI) model. There are four kinds of agents in the layer depicted in Fig. 2.

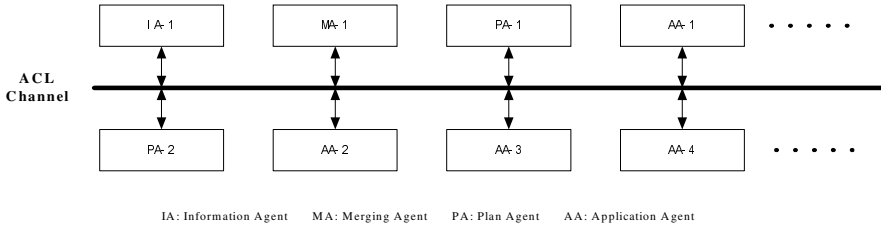


Fig. 2. Multi-agent platform structure

5.1 Plan Agent (PA)

PA is the main component of abstract workflow. It is domain specific, and its capability is always stable and predefined. In some sense, PA takes the role of host of sub-workflow denoted in business workflow domain.

PA plan behaves externally as three parts:(1)Goal(Post-conditions of the plan);(2)Context(Pre-conditions of the plan);(3)Content(Actions' sequence of the plan). To find a suitable plan only means to search in the plan library to get the plan whose context can be meet by the given conditions and goal can contain the required intention. PA plan can be built either manually by agent designer using the micro-workflow modeling tool, or automatically by the planner using *Algorithm 1*. Each plan has many common components comparing with the generic workflow, such as logical nodes, internal actions (*start, finish, delay, evaluate, convert*). But external actions, such as invocation of Web or Grid Services, operations on database, or execution of other kinds of application, will be replaced with application agents (AAs) in the plan. We prefer no other plans embedded in PA's plan. Nested plans will be complicated. So the actions in the plan of PA are all atomic type, and they can be executed directly. The capability of a PA may be represented with the intersection of preconditions and the conjunction of effects of its all plans. Some definitions are given below:

$$Plans(PA) = \{plan_1, plan_2, \dots, plan_n\}$$

$$Capability(PA) = \{Preconditions(PA), Effects(PA)\}$$

$$Preconditions(PA) = Preconditions(plan_1) \cap Preconditions(plan_2) \cap \dots \cap Preconditions(plan_n)$$

$$Effects(PA) = Effects(plan_1) \cup Effects(plan_2) \cup \dots \cup Effects(plan_n)$$

The preconditions of PA are seen as a part of PA's beliefs, and the effects of PA are seen as PA's intention. To check whether a PA has the demanded capability means to compare their preconditions and effects. If the two qualifications given below could be fulfilled, the PA will have the demanded capability.

- (1) $\text{Preconditions(PA)} \subseteq \text{Preconditions(given)}$
 (2) $\text{Effects(PA)} \supseteq \text{Effects(needed)}$

Due to above definitions, such decision is loose. The satisfied PA might have no eligible plan to achieve the demanded goal. In the situation, a new plan should be generated automatically and the PA's capability will be updated and enhanced.

5.2 Application Agent (AA)

Each AA corresponds some application adapters from the same domain. Both preconditions and effects of its capability are the conjunctive ones of its all application adapter instances. Some definitions are given below:

$$\text{Adapters(AA)} = \{ \text{adapter}_1, \text{adapter}_2, \dots, \text{adapter}_n \}$$

$$\text{Capability(AA)} = \{ \text{Preconditions(AA)}, \text{Effects(AA)} \}$$

$$\text{Preconditions(AA)} = \text{Preconditions}(\text{adapter}_1) \cup \text{Preconditions}(\text{adapter}_2) \cup \dots \cup \text{Preconditions}(\text{adapter}_n)$$

$$\text{Effects(AA)} = \text{Effects}(\text{adapter}_1) \cup \text{Effects}(\text{adapter}_2) \cup \dots \cup \text{Effects}(\text{adapter}_n)$$

During the generation of a new plan, PA will search for suitable AAs in action library that includes all relevant AAs according to the demanded initial states and goals. The two qualifications are given below:

- (1) $\text{Preconditions(AA)} \subseteq \text{Preconditions(given)}$
 (2) $\text{Effects(AA)} \supseteq \text{Effects(needed)}$

IO parameters of those adapter instances are also stored in AA. According to different application types, AAs have different names: Service agent (SAG), Java Class agent (JCAg), Database agent (DBAg), Command Line agent (CLAg), etc. SAG corresponds a list of web services that have the specific function. In planning phase, SAG is responsible for binding to the best service, which meets the action requirements of precondition and effect and has the maximum QoS value. In executing phase, SAG will invoke the selected service. So SAG is the intelligent and adaptive service broker.

5.3 Information Agent (IA)

IA is the registry of all agents in the multi-agent platform. We adopt multi-layer distributed framework for IA in Grid environment. Each VO has its local IA. The local IA keeps the information about capabilities of all PAs and AAs in the VO. AAs are chosen preferentially from the same VO when building a plan in order to get better efficiency in time of processing. The general IA has the registration information about all local IAs.

5.4 Merging Agent (MA)

MA is responsible for merging all related plans of PAs in a user-defined abstract workflow to get a concrete workflow. Both POP and HTN [14] are utilized by MA. Details are given in *Algorithm 2*.

6 A Case Study and the Implementation

In order to show how the planning technology presented in the paper works, this section introduces a service-based system model for online conference arrangement planning, which needs to integrate services provided by different organizations such as conference organizers, ticket agents and hotels. We need build domain-specific ontologies for planning in advance. Plan generation and selection, PA selection, AA selection and applications adapter selection are all based on those term ontologies.

6.1 Abstract Workflow Model

We assume several international conferences will be held in Shanghai Jiao Tong University this year, so the conference organizers want to build an online conference arrangement service for the future attendee. At first step, they build an abstract workflow using the macro-workflow modeling tool (See Fig. 3).

With their experiences, they believe some parameters should be given before invoking the workflow. We may call these parameters “Preconditions”(See Table 1).

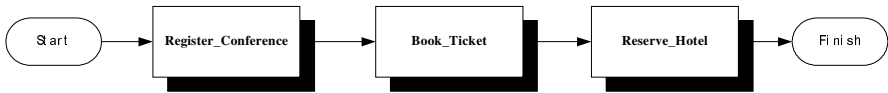


Fig. 3. Abstract workflow model

Table 1. Preconditions of the workflow

Name	Description
<i>Conf_name</i>	Conference name
<i>Attendee_name</i>	Attendee name
<i>Attendee_city</i>	City attendee lives
<i>Vehicle_type</i>	Vehicle type (Plane, Train, Bus, Ship)
<i>Hotel_level</i>	Level of hotel

The organizers also define the requirements of each functionality module in the model (See Table 2).

Table 2. Requirements of functionality modules

Module	Precondition	Effect	Description
<i>Register_Conference</i>	<i>Conf_name</i> <i>Attendee_name</i>	<i>Conf_starttime</i> <i>Conf_endtime</i> <i>Conf_city</i> <i>Registration_id</i>	Provide conference registration service
<i>Book_Ticket</i>	<i>Source_city</i> <i>Destination_city</i> <i>Start_date</i> <i>Passenger_name</i> <i>Vehicle_type</i>	<i>Vehicle_no</i> <i>Ticket_cost</i> <i>Start_time</i> <i>Arrival_time</i> <i>Booking_id</i>	Book the return ticket from the conference city
<i>Reserve_Hotel</i>	<i>Conf_city</i> <i>Attendee_name</i> <i>Hotel_level</i> <i>Enter_date</i> <i>Leave_date</i>	<i>Hotel_name</i> <i>Hotel_address</i> <i>Reservation_id</i> <i>Duration</i> <i>Total_cost</i>	Reserve the required hotel for attendee

6.2 Present Plan Agents

Three plan agents are assumed in the system. Their descriptions are given below.

- **Conference_Registration PA:**

The *conference_Registration* PA has only one plan in its plan library. It is *CRI*.

```

Action( ConfInquiry, PRECOND: Conf_name, EFFECT: Conf_starttime ^ Conf_endtime ^ Conf_city)
Action( AttendeeRegistration, PRECOND: Conf_name ^ Attendee_name,
      EFFECT: Registration_id ^ Conf_schedule)
Decompose(Conference_Registration,
Plan(STEPS: {S1: ConfInquiry, S2: AttendeeRegistration}
ORDERINGS: {Start < S1 < Finish, Start < S2 < Finish}
LINKS: {Start  $\xrightarrow{\text{Conf\_name}}$  S1, S1  $\xrightarrow{\text{Conf\_starttime, Conf\_endtime, Conf\_city}}$  Finish,
      Start  $\xrightarrow{\text{Conf\_name, Attendee\_name}}$  S2, S2  $\xrightarrow{\text{Registration\_id, Conf\_schedule}}$  Finish})
    
```

And the capability of the PA may be represented as below:

```

Capability( Conference_Registration, PRECOND: Conf_name ^ Attendee_name,
      EFFECT: Conf_starttime ^ Conf_endtime ^ Conf_city ^ Registration_id ^ Conf_schedule)
    
```

- **Ticket_Booking PA:**

The *Ticket_Booking* PA has two plans: *TB1*, *TB2*. Definition of *TB1* is:

```

Action(TicketQuery1, PRECOND: Source_city ^ Destination_city ^ Start_date ^ Time_of_period,
      EFFECT: Vehicle_no ^ Ticket_cost ^ Start_time ^ Arrival_time)
Action(PassengerBooking, PRECOND: Start_date ^ Vehicle_no ^ Passenger_name, EFFECT: Booking_id)
Decompose(Ticket_Booking,
Plan(STEPS: {S1: TicketQuery1, S2: PassengerBooking}
ORDERINGS: {Start < S1 < S2 < Finish}
LINKS: {Start  $\xrightarrow{\text{Source\_city, Destination\_city, Start\_date, Time\_of\_period}}$  S1, S1  $\xrightarrow{\text{Vehicle\_no}}$  S2,
      S2  $\xrightarrow{\text{Booking\_id}}$  Finish})
    
```

Definition of *TB2* is:

```

Action(TicketQuery2, PRECOND: Source_city ^ Destination_city ^ Start_date ^ Vehicle_type,
      EFFECT: Vehicle_no ^ Ticket_cost ^ Start_time)
Action(PassengerBooking, PRECOND: Start_date ^ Vehicle_no ^ Passenger_name, EFFECT: Booking_id)
Decompose(Ticket_Booking,
Plan(STEPS: {S1: TicketQuery2, S2: PassengerBooking}
ORDERINGS: {Start < S1 < S2 < Finish}
    
```

```

LINKS: {Start  $\xrightarrow{\text{Source\_city, Destination\_city, Start\_date, Time\_of\_period}}$  S1, S1  $\xrightarrow{\text{Vehicle\_no}}$  S2,
      S2  $\xrightarrow{\text{Booking\_id}}$  Finish})
    
```

And the capability of the PA may be represented as below:

```

Capability(Ticket_Booking, PRECOND: Source_city ^ Destination_city ^ Start_date ^ Passenger_name,
      EFFECT: Vehicle_no ^ Ticket_cost ^ Start_time ^ Arrival_time ^ Booking_id)
    
```

- **Hotel_Reservation PA:**

We assume that every attendee will reserve a standard room. The *Hotel_Reservation* PA has one plan: *HRI*. Definition of *HRI* is:

```

Action(HotelSearch, PRECOND: Conf_city ^ Hotel_level, EFFECT: Hotel_name ^ Hotel_address)
Action(GuestReservation, PRECOND: Hotel_name ^ Attendee_name ^ EnterDate ^ LeaveDate,
      EFFECT: Reservation_id ^ Duration ^ Total_cost)
Decompose(Hotel_Reservation,
Plan(STEPS: {S1: HotelSearch, S2: GuestReservation}
ORDERINGS: {Start < S1 < S2 < Finish}
LINKS: {Start  $\xrightarrow{\text{Conf\_city, Hotel\_level}}$  S1, S1  $\xrightarrow{\text{Hotel\_name}}$  S2,
      S2  $\xrightarrow{\text{Reservation\_id, Duration, Total\_cost}}$  Finish})
    
```

And the capability of the PA may be represented as below:

Capability(*Hotel_Reservation*,
 PRECOND: *Conf_city* \wedge *Attendee_name* \wedge *Hotel_level* \wedge *Enter_date* \wedge *Leave_date*,
 EFFECT: *Hotel_name* \wedge *Hotel_address* \wedge *Reservation_id* \wedge *Duration* \wedge *Total_cost*)

6.3 Analysis and Planning

We assume the attendee will arrive at the day before the conference open day and go back at the day after the conference end day. The assumption adds some exact relationships among those conditions and simplifies the problem solving. According to the user requirements about three functionality modules, appropriate PAs are selected and matched at first. Then we get the results as follows:

- *Conference_Registration* PA can satisfy the requirements of *Register_Conference* module, and *CR1* plan is selected.
- *Ticket_Booking* PA can satisfy the requirements of *Book_Ticket* module, but *TB1* and *TB2* in its plan library are all not befitting. With the new preconditions and effects, a suitable plan *TB3* is generated using *Algorithm 1*:

Action(*TicketQuery3*, PRECOND: *Source_city* \wedge *Destination_city* \wedge *Start_date* \wedge *Vehicle_type*,
 EFFECT: *Vehicle_no* \wedge *Ticket_cost* \wedge *Start_time* \wedge *Arrival_time*)
 Action(*PassengerBooking*, PRECOND: *Start_date* \wedge *Vehicle_no* \wedge *Passenger_name*, EFFECT: *Booking_id*)
 Decompose(*Ticket_Booking*,
 Plan(STEPS: {*S1*: *TicketQuery3*, *S2*: *PassengerBooking* }
 ORDERINGS: {*Start* < *S1* < *S2* < *Finish* }
 LINKS: {*Start* $\xrightarrow{\text{Source_city, Destination_city, Start_date, Time of period}}$ *S1*, *S1* $\xrightarrow{\text{Vehicle_no}}$ *S2*,
S2 $\xrightarrow{\text{Booking_id}}$ *Finish*})

The capability of the PA also holds the line.

- *Hotel_Reservation* PA can satisfy the requirements of *Reserve_Hotel* module, and *HR1* plan is selected.

6.4 Final Concrete Workflow

At last, the final concrete workflow is generated by MA using *Algorithm 2*. It is depicted in Fig. 4. For the six AAs in the final workflow: *ConfInquiry*、*AttendeeRegistration*、*TicketQuery3*、*PassengerBooking*、*HotelSearch* and

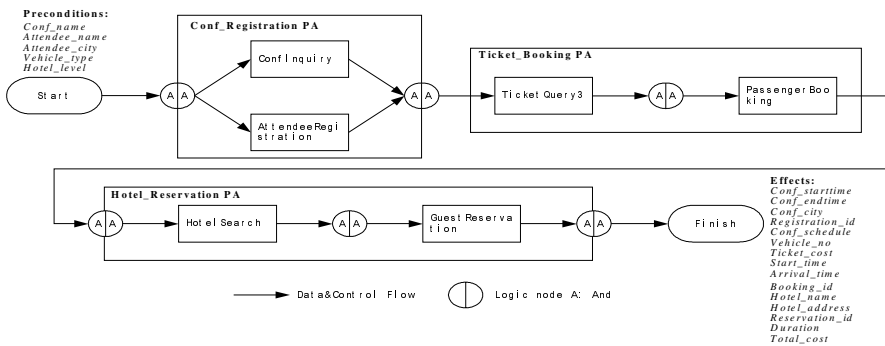


Fig. 4. Final concrete workflow

GuestReservation, each of them will choose the best adapter instance having the maximum QoS value from the candidates, and bind it to run.

6.5 System Prototype Implementation

Our AGWMS is developed with Java language. A prototype has been developed according to the framework. Fig. 5(a) gives a screenshot of the macro-workflow modeling tool. Fig. 5(b) shows web service adapter interface. Our multi-agent platform is developed using JADE [16] technology. Fig. 5(c) shows the agent-hosting environment. The prototype runs on Windows platform.

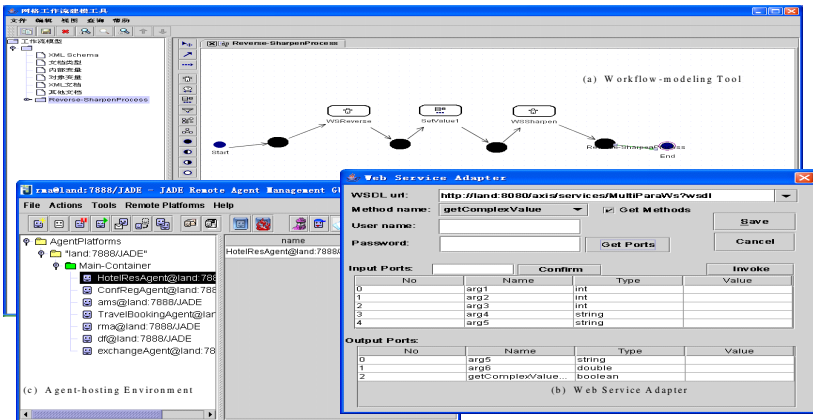


Fig. 5. System prototype

7 Conclusion and Future Work

This paper has described a planning enhanced Grid Workflow Management System based on agent (AGWMS). We give its four-layer architecture and describe its components. The adapter middleware increases the scalability of AGWMS. Use of agent technology makes AGWMS more intelligent. Our multi-agent platform makes the system meet challenges from the grid context. AI planning technology is adopted to enhance the intelligence and robustness of the system. In the near future, we plan to go further using planning technology in AGWMS and do a practical performance evaluation to show that our agent-based workflow management system can be efficiently used in the Grid environment.

References

- [1] I. Foster and C. Kesselman (editors), *The Grid Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann Publishers, USA, 1999.
- [2] Soonwook Hwang, *Grid workflow-A flexible framework for fault tolerance in the grid*, Phd Thesis

- [3] Ewa Deelman, James Blythe, Yolanda Gil, and Carl Kesselman, "Workflow Management in GriPhyN", *The Grid Resource Management*, Kluwer 2003
- [4] Jini Community. <http://www.jini.org/>
- [5] Condor Team, *The directed acyclic graph manager*, <http://www.cs.wisc.edu/condor/dagman>, 2004
- [6] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao, *Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation*, presented at Scientific and Statistical Database Management, 2002
- [7] Ewa Deelman, et al., *Pegasus : Mapping Scientific Workflows onto the Grid*, Across Grids Conference 2004, Nicosia, Cyprus
- [8] MyGrid Project. <http://www.mygrid.org.uk/>
- [9] J. Cao, S. A. Jarvis, S. Saini, G. R. Nudd. *GridFlow: Workflow Management for Grid Computing*. In Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03), 2003, 198-205
- [10] G.E. Graham, D. Evans and I. Bertram: *McRunjob: A High Energy Physics Workflow Planner for Grid Computing in High Energy and Nuclear Physics*. La Jolla, California 3(2003)
- [11] Hugh P. Bivens and Judy I. Beiriger: *GALE: Grid Access Language for HPC Environments*, Available at ass3186.sandia.gov/hpbiven.
- [12] BPEL4WS Specification, <http://www-106.ibm.com/developerworks/library/ws-bpel/>
- [13] S. Tuecke, e.a.: *Open grid services infrastructure (OGSI) version 1.0*. In: Specification of GGF by OGSi-WG. (2003)
- [14] Stuart Russell, Peter Norvig, *Artificial Intelligence: A Modern Approach (Second Edition)*, Prentice Hall, 2002
- [15] Michael Wooldridge, *Introduction to MultiAgent Systems*, John Wiley and Sons, 2002
- [16] JADE project, <http://jade.tilab.com/>

Reasoning Based on the Distributed β -PSML

Yila Su^{1,2}, Jiming Liu³, Ning Zhong⁴, and Chunnian Liu¹

¹ College of Computer Science, Beijing University of Technology, China

² College of Information Engineering,

Inner Mongolia University of Technology, China

³ Department of Computer Science, Hong Kong Baptist University, Hong Kong

⁴ Department of Information Engineering, Maebashi Institute of Technology, Japan
syl@emails.bjut.edu.cn, jiming@comp.hkbu.edu.hk,
zhong@maebashi-it.ac.jp, ai@bjut.edu.cn

Abstract. Information on the Web can be either globally distributed throughout the Web within multi-layer over the infrastructure of Web protocols, or located locally, portal-centralized Web services (i.e., the intelligent service provider) that is integrated to its own cluster of specialized intelligent applications. However, each approach has a serious flaw. We propose the Distributed β -PSML (Problem Solver Markup Language) for solving problems in a large-scale distributed Web environment by combining global and local knowledge-data bases in this paper. The Distributed β -PSML is an extension of the β -PSML that we proposed before for complex adaptive, distributed problem solving, and can be easily used for automatic reasoning on the Web for decision-making and e-business intelligence. We also give an example to show how the Distributed β -PSML can be used for automatic reasoning on the Web by incorporating global information sources from the Semantic Web with locally operational knowledge-data bases in an enterprise portal.

1 Introduction

Information on the Web can be either globally distributed throughout the Web within multi-layer over the infrastructure of Web protocols, or located locally, portal-centralized Web services (i.e., the intelligent service provider) that is integrated to its own cluster of specialized intelligent applications. However, each approach has a serious flaw. We propose the Distributed β -PSML (Problem Solver Markup Language) for solving problems in a large-scale distributed Web environment by combining global and local knowledge-data bases in this paper. The Distributed β -PSML is an extension of the β -PSML that we proposed before for complex adaptive, distributed problem solving, and can be easily used for automatic reasoning on the Web for decision-making and e-business intelligence. We also give an example to show how the Distributed β -PSML can be used for automatic reasoning on the Web by incorporating global information sources from the Semantic Web with locally operational knowledge-data bases in an enterprise portal.

The remaining of this paper is as follows. Section 2 presents how to construct the Distributed β -PSML and the reasoning mechanism of the Distributed

β -PSML. Section 3 provides experimental results to show usefulness of our approach. Section 4 surveys related work. Finally, Section 5 gives concluding remarks.

2 The Framework of the Distributed β -PSML

We propose the Distributed β -PSML for solving problems in a large-scale distributed Web environment by combining global and local knowledge-data bases.

2.1 The Web Structure Based on the Distributed β -PSML

As elaborated based on [11], distributed problem solving on the Web will provide the following functions:

- The expressive power and functional support in PSML for complex adaptive, distributed problem solving;
- Performing automatic reasoning on the Web by incorporating globally distributed contents and meta-knowledge automatically collected and transformed from the Semantic Web and social networks with locally operational knowledge-data bases;
- Representing and organizing multiple, huge knowledge-data sources for distributed Web inference and reasoning;
- Combining multiple reasoning methods in the method efficiently and effectively;
- Modeling user behavior and representing/managing it as a personalized model dynamically.

Figure 1 shows the Web structure Based on the Distributed β -PSML. In fact, when Distributed β -PSML is considered, its syntax and semantics will become much more complex than that of the β -PSML [11]. In this case, a query described with the Distributed β -PSML will consist of two parts: one part is to describe the query, and the other part presents constraints that will be useful to answer the query. To express our idea clearly, we turn to introduce the structure of the nodes on the Web in distributed environment. Each node has an inference engine and a knowledge-base. The knowledge-base consists of three parts: meta-knowledge, constraint knowledge and domain knowledge, respectively. Here meta-knowledge is composed of experience knowledge and heuristic knowledge and is used for classifying the query and dividing the query into sub-queries.

As shown in Figure 2, the dynamic information/knowledge on the Web can be organized on the Grid with multi-service levels [2]. In order to achieve a goal by making use of some services-oriented distributed resources, problem solving related techniques are commonly used. After relevant services are identified, the compatible ones can then be composed to form a service flow for subsequent execution. We here argue that a hierarchy of services is needed to support the corresponding service flow process. At the lowest layer, the Grid fabric can provide a platform that enables **resources** sharing in dynamic and distributed virtual

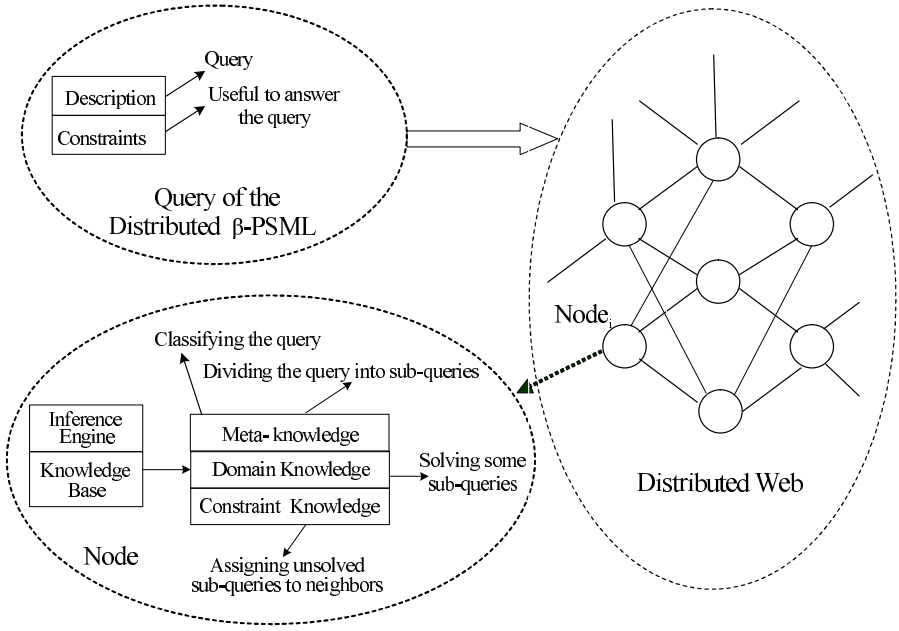


Fig. 1. The Web structure Based on the Distributed β -PSML

organizations. Embracing Web **services** on Grid has recently been realized as an Open Grid Services Architecture (OGSA), which opens up the service oriented computing paradigm to be applied in Grid. Every heterogeneous resource is virtualized as network-enabled Grid Services. With the use of Semantic Web technology, Web Services can be described **semantically** and make service discovery and matchmaking based on service semantics and ontologies possible. The semantics used for tagging the services should further be organized to constitute **knowledge** about services. With the assumption that the service knowledge is distributed on the Web and owned by different agents with different social and trust relationships, the **social network** connecting the corresponding agents becomes an inevitable layer that can facilitate dynamic knowledge exchange and sharing.

2.2 The Reasoning Process of the Distributed β -PSML

When a node in the distributed environment receives a query, the node classifies the query or divides the query into sub-queries with its meta-knowledge, solves some of them with its domain knowledge, and assigns unsolved queries to its neighbors with its constraint knowledge. Just like a social network, the connection between two nodes is formed by accumulated experience. The mechanism of assigning sub-queries to the neighbors of a node is similar that of case-based reasoning. Each neighbor handles the sub-queries in the similar way and the

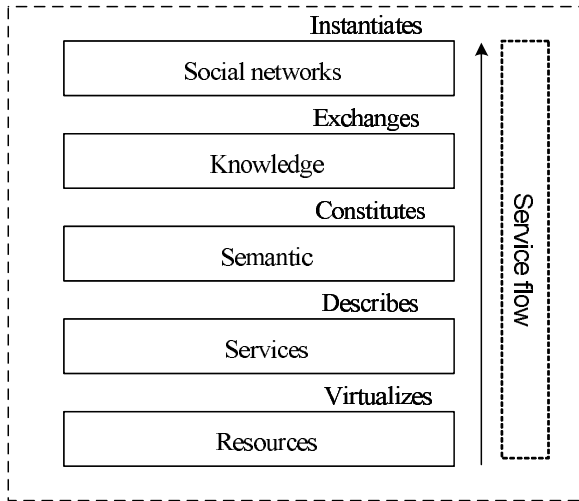


Fig. 2. The Web with multi-service levels

process continues until the node solves all received sub-queries. Then it returns the sub-answers recursively. When all the neighbors of the original node return answers of the received sub-queries, it combines them into the answer of the original query and returns it to the user. Each node can use an inference method which can be different from others (e.g. some nodes use exact inference methods, while some others use uncertainty inference). The answer may be incomplete or empty because some sub-queries cannot be transferred to the neighbors of the node according to the constraint knowledge, which means no neighbor can solve the sub-queries or the constraints of the sub-queries and the node can not be satisfied [11].

3 Experiments

Experiments in this section show how the Distributed β -PSML can be used for automatic reasoning on the Web by incorporating global information sources from the Semantic Web with locally operational knowledge-data bases in an enterprise portal.

3.1 A Camera Ontology for a Camera-Sale Portal

We use a sample of camera ontology to demonstrate our approach as shown in Figure 3. In this figure, the plain line denotes that the subclass relation holds between the connected nodes; the dashed line denotes slot of attribute and set of attributes; the circle object denotes a class; and the rectangle denotes an attribute owned by the class.

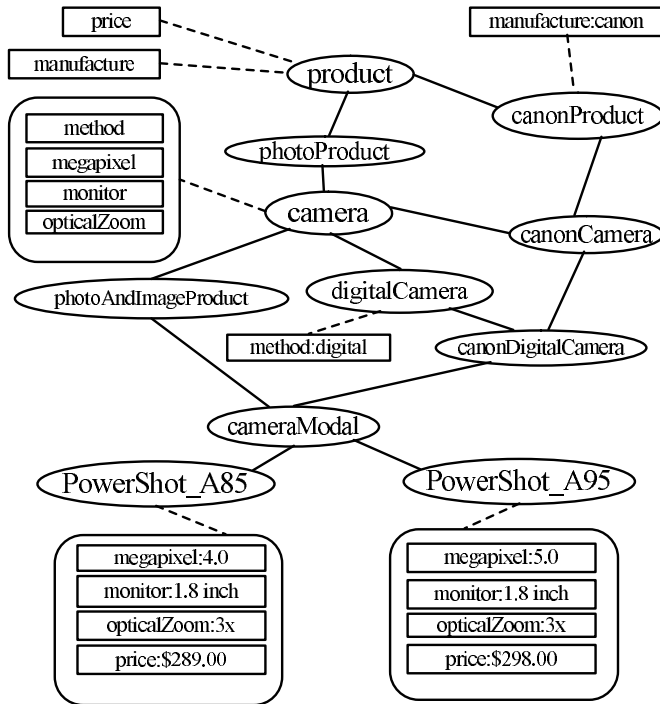


Fig. 3. A sample of camera ontology

A Camera-Sale Portal

CameraPortal > Sony > DSC-W1

<p>SAMSUNG</p> <ul style="list-style-type: none"> • V50 5MP • 360 3.1MP • 210 SE 2MP • 410 4MP • 130 1.3MP <p>Sony</p> <ul style="list-style-type: none"> • DSC-W1 • DSC-T1 • DSC-P100 • DSC-P93 • DSC-P41 • DSC-P73 • DSC-F88 • DSC-V1 • DSC-U60 • DSC-U40 • DSC-F828 	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td colspan="2" style="text-align: center;">Sony Cybershot DSC-W1 5MP Digital Camera with 3x Optical Zoom</td> </tr> <tr> <td colspan="2" style="text-align: center;">www.amazon.com</td> </tr> <tr> <td colspan="2" style="text-align: center;">\$399.87</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2" style="text-align: center;">Technical Data</th> </tr> </thead> <tbody> <tr> <td>Resolution mode</td> <td>2592 x 1944 (5.1MP)</td> </tr> <tr> <td>LCD monitor</td> <td>2.5-inch</td> </tr> <tr> <td>Optical Zoom</td> <td>3x</td> </tr> <tr> <td rowspan="2">Movie Mode</td> <td>frames per second</td> <td>30</td> </tr> <tr> <td>VGA</td> <td>640 x 480</td> </tr> <tr> <td>Size (W x H x D)</td> <td>3.56 x 2.38 x 1.25 inches</td> </tr> <tr> <td>Power</td> <td>NiMH AA-sized batteries (2100 mAh)</td> </tr> <tr> <td>Storage</td> <td>32 MB - 1 gigabyte</td> </tr> <tr> <td>Transfer</td> <td>USB 2.0</td> </tr> <tr> <td colspan="2">Real Imaging Processor, Selectable Focus Mode, 5 Area Multi-Point Auto Focus, AF Illuminator, Multi-Pattern Measuring, Multi-Burst Mode, Manual Exposure Mode</td> </tr> </tbody> </table>	Sony Cybershot DSC-W1 5MP Digital Camera with 3x Optical Zoom		www.amazon.com		\$399.87		Technical Data		Resolution mode	2592 x 1944 (5.1MP)	LCD monitor	2.5-inch	Optical Zoom	3x	Movie Mode	frames per second	30	VGA	640 x 480	Size (W x H x D)	3.56 x 2.38 x 1.25 inches	Power	NiMH AA-sized batteries (2100 mAh)	Storage	32 MB - 1 gigabyte	Transfer	USB 2.0	Real Imaging Processor, Selectable Focus Mode, 5 Area Multi-Point Auto Focus, AF Illuminator, Multi-Pattern Measuring, Multi-Burst Mode, Manual Exposure Mode	
Sony Cybershot DSC-W1 5MP Digital Camera with 3x Optical Zoom																														
www.amazon.com																														
\$399.87																														
Technical Data																														
Resolution mode	2592 x 1944 (5.1MP)																													
LCD monitor	2.5-inch																													
Optical Zoom	3x																													
Movie Mode	frames per second	30																												
	VGA	640 x 480																												
Size (W x H x D)	3.56 x 2.38 x 1.25 inches																													
Power	NiMH AA-sized batteries (2100 mAh)																													
Storage	32 MB - 1 gigabyte																													
Transfer	USB 2.0																													
Real Imaging Processor, Selectable Focus Mode, 5 Area Multi-Point Auto Focus, AF Illuminator, Multi-Pattern Measuring, Multi-Burst Mode, Manual Exposure Mode																														

Fig. 4. The sample of camera-sale portal

Furthermore, the camera ontology is utilized to develop a camera-sale portal on the Semantic Web. Figure 4 shows the Web browser of this portal. In the camera-sale portal, we use RDF-Schema format for instances and the camera ontology in OWL that builds upon the RDF and RDF Schema, so that XSLT can be employed to transform Web contents information on the Semantic Web (global information sources) [3].

3.2 Workflow of Transformation

Figure 5 shows a workflow of transformation from Web-contents information on the Semantic Web into the Distributed β -PSML [12]. As discussed in [11], ontologies represented in OWL can be easily translated to β -PSML. The transformation process is the XSLT engine for analyzing an input source with ontologies (e.g. XML, RDF, and OWL) and for transforming it into the Distributed β -PSML as an output source. The examples of the input and output sources are shown in Figures 6-9, respectively..

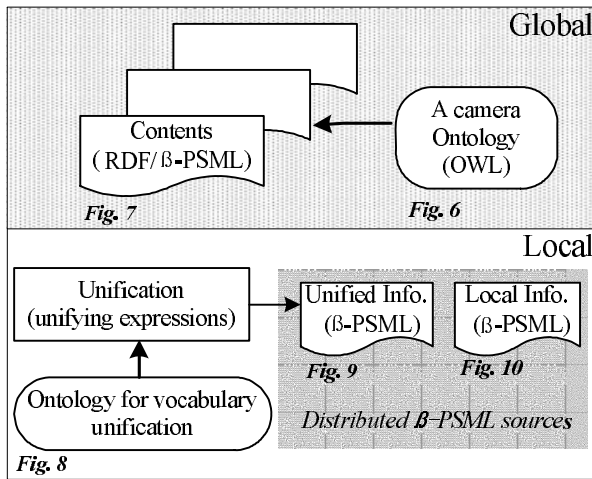


Fig. 5. Workflow of transformation

3.3 Inference and Reasoning of the Distributed β -PSML in Experiments

In this subsection, we describe the inference and reasoning process of the Distributed β -PSML that are performed by transforming the global Semantic Web information sources and coupling with knowledge-data bases stored in the local machine in advance. Part of the sample knowledge-data base in the local machine is shown in Figure 10. The knowledge-data base stores the information of the company itself and the old information of other companies.

The collected information is defined in the maverick style. Here are some problems: we must unify documentary form of the collected information because all manufacturers do not use the same format. Of course, if all manufacturers will use the unified format for contents, it will be easy to solve this problem. Unfortunately, it is difficult to ask for all manufacturers to use the same format.

Because the unification process of XML hierarchy is very difficult, we assumed that XML hierarchy is the same, but the used vocabularies may be different. We developed Unification for unifying vocabularies contained in Web contents information.

The workflow of Unification can be described as follows. First, Unification reads and analyzes the camera ontology denoted in OWL (see Figure 6) and instances denoted in RDF (see Figure 7). Secondly, Unification calls the XSLT engine and reads the ontology for vocabulary unification (see Figure 8). Finally, the output of Unification is the Distributed β -PSML file (see Figure 9) with a unified vocabulary.

```

<owl:Class rdf:ID="product">
  <rdfs:subClassOf>
    <rdfs:Restriction>
      <owl:onProperty rdf:resource="#price"/>
    </rdfs:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <rdfs:Restriction>
      <owl:onProperty rdf:resource="#manufacturer"/>
    </rdfs:Restriction>
  </rdfs:subClassOf>
</owl:Class> <owl:Class rdf:ID="photoProduct">
  <rdfs:subClassOf rdf:resource="#product"/>
</owl:Class>

```

Fig. 6. Part of the camera ontology in OWL

More specifically, the transformation collects Web contents information on the Semantic Web by Web information collecting agents. However, Web content information may not be afforded by camera manufacturer. For this reason, we utilized contents information (XML, OWL, the Distributed β -PSML) of our own which was drawn upon information of camera manufacturer's Web sites. These will be expected to be afforded by all manufacturers in the future.

We got ready for query by using the predicate *find* for the purpose of inferring by using both local and global information sources. It can enumerate products by matching the criteria specified. Figure 11 shows the results with respect to query-answering by using *find* in PSML.

In Figure 11, “*Camera*” is a variable of camera products. “*megapixel (\$ge, 3.5)*” is an assumption about ‘megapixel is over 3.5’, that means ‘enumerate cameras which the megapixel is over 3.5’. When this query is given, the Distributed β -PSML reasons with global and local sources, and then gives answers about products that are acceptable. That is, the reasoning is carried out on the camera ontology and instances transformed from the Semantic Web with locally cameras related operational knowledge-data bases.

```

<rdf:Description rdf:ID=" PowerShot A85">
  <rdf:type rdf:resource=
    "http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="&camera;cameraModel"/>
  <camera:name>
    <xsd:string>
      <rdf:value>PowerShot A85</rdf:value>
    </xsd:string>
  </camera:name>
  <camera:megapixel>
    <xsd:string>
      <rdf:value>4.0</rdf:value>
    </xsd:string>
  </camera: megapixel>
  <camera:monitor>
    <xsd:string>
      <rdf:value>1.8 inch</rdf:value>
    </xsd:string>
  </camera:monitor >
</rdf:Description>

```

Fig. 7. Description of instances in RDF

```

<owlx :Class rdf:ID="name">
  <rdfs:seeAlso rdf:resource="#product"/>
  <rdfs:seeAlso rdf:resource="#product_name"/>
</owlx:Class> <owlx:Class rdf:ID="megapixel">
  <rdfs:seeAlso rdf:resource="#mega_pixel"/>
  <rdfs:seeAlso rdf:resource="#resolution_mode"/>
  <rdfs:seeAlso rdf:resource="#pixel"/>
</owlx:Class> <owl:Class rdf:ID="storage">
  <rdfs:seeAlso rdf:resource="#memory_stick"/>
  <rdfs:seeAlso rdf:resource="#memory"/>
</owlx:Class>

```

Fig. 8. A sample ontology for vocabulary unification in OWL

If only local sources are employed for inference, the answers are incomplete (Figure 12). Hence, combining global and local information sources is very important for decision-making based on the Web inference engine.

4 Related Work

There have been several works addressing the combining OWL with Horn clauses or other logic systems for problem solving and rule markup languages on the Semantic Web. They include SWRL [7], XRML [8], and the W4 project [1], UPML [10], and information retrieval on the Semantic Web documents [4].

SWRL constructed a Semantic Web Rule Language by combining OWL and RuleML, but its number of predicate arguments was limited to only one or two and also its reasoning mechanism was not considered. In [8], Kang and Lee observed that there exist rules embedded implicitly in Web pages that cannot be processed in XML, and proposed XRML that supports the automatic processing of implicit rules embedded in the hypertexts and helps human's browsing for their comprehension. However, XRML only utilizes global information and does not consider the ontology. The W4 (Well-Founded Semantics for the WWW)


```

<prolog:clause>
  <prolog:head>
    <owls:Class rdf:ID="canonProduct">
      <rdfs:subClassOf>
        <owls:Restriction>
          <owls:hasValue rdf:datatype=
            "&xsd:string">canon</owls:hasValue>
          <owls:onProperty rdf:resource="manufacturer"/>
        </owls:Restriction>
      </rdfs:subClassOf>
    </owls:Class>
    <owls:Class rdf:ID="product"/>
    </owls:Class>
    <owls:Class rdf:ID="digitalCamera">
      <rdfs:subClassOf>
        <owls:Restriction>
          <owls:hasValue rdf:datatype=
            "&xsd:string">digital</owls:hasValue>
          <owls:onProperty rdf:resource="method"/>
        </owls:Restriction>
      </rdfs:subClassOf>
    </owls:Class>
    <owls:Class rdf:ID="camera"/>
    </owls:Class>
    <owls:Class rdf:ID="photoProduct">
      <rdfs:subClassOf rdf:resource="#product"/>
    </owls:Class>
    <owls:Class rdf:ID="photoAndImageProduct">
      <rdfs:subClassOf rdf:resource="#camera"/>
    </owls:Class>
    <owls:Class rdf:ID="cameraModel">
      <rdfs:subClassOf rdf:resource=
        "#photoAndImageProduc"/>
    </owls:Class>
    ...
    <cameraModel rdf:ID="PowerShot_A95">
      <price rdf:datatype="&xsd:string">$398.00</price>
      <opticalZoom rdf:datatype=
        "&xsd:string">3x</opticalZoom>
      <megapixel rdf:datatype="&xsd:string">5.0</megapixel>
      <monitor rdf:datatype="&xsd:string">1.8 inch</monitor>
    </cameraModel>
  </prolog:head>
</prolog:clause>

```

Fig. 9. The sample of camera ontology transformed to the Distributed β -PSML

project aims at developing Standard Prolog inter-operable tools for supporting distributed, secure, and integrated reasoning activities on the Semantic Web, but it did not also consider the ontology. Compared with the work mentioned above, our research concentrates on the combination of OWL with Horn clauses, and the PSML provides capabilities to represent, transform and manage multiple, huge knowledge-data sources on the Web and the knowledge grid for distributed Web inference and reasoning as well as complex adaptive, distributed problem solving.

UPML was developed for unifying problem solving methods that provides reusable architectures and components for implementing the reasoning part of knowledge based systems. UPML can describe architectures for reasoning of deeper knowledge. One of major differences with our work is concerning with the

```

<cameraModel rdf:ID="DSC-W1">
  <price rdf:datatype="xsd:string">$387.00</price>
  <opticalZoom rdf:datatype="xsd:string">3x</opticalZoom>
  <megapixel rdf:datatype="xsd:string">5.26</megapixel>
  <monitor rdf:datatype="xsd:string">2.5 inch</monitor>
</cameraModel> <cameraModel rdf:ID="DSC- T1">
  <price rdf:datatype="xsd:string">$589.00</price>
  <opticalZoom rdf:datatype="xsd:string">3x</opticalZoom>
  <megapixel rdf:datatype="xsd:string">5.1</megapixel>
  <monitor rdf:datatype="xsd:string">2.5 inch</monitor>
</cameraModel> <cameraModel rdf:ID="DSC- P72">
  <price rdf:datatype="xsd:string">$254.00</price>
  <opticalZoom rdf:datatype="xsd:string">3x</opticalZoom>
  <megapixel rdf:datatype="xsd:string">3.2</megapixel>
  <monitor rdf:datatype="xsd:string">1.5 inch</monitor>
</cameraModel>

```

Fig. 10. Part of the local knowledge-data base

markup language. UPML utilized a specific markup language, but our research is based on a standard markup language on the Semantic Web, such as OWL, RDF, etc. Hence, our approach can deal with various information sources on the Semantic Web.

In [10], Quaresma and Rodrigues developed an information retrieval system with two layers, the information retrieval layer and the interaction layer, that uses the Semantic Web for improvement in quality. However, our objectives are not only focusing on retrieval, but also for providing knowledge services including decision-making on intelligent enterprise portals.

5 Conclusions

The Distributed β -PSML that we proposed is a new method of knowledge representation for the Semantic Web. The Distributed β -PSML can be easily used for inference and reasoning as well as transforming and managing both global and local knowledge-data bases. The Distributed β -PSML can perform the function of logic layer for the Semantic Web. The experimental results show that our considerations are valid and our preliminary solution works well on the Semantic Web.

In the future work, we will consider how to use the knowledge with a hierarchical structure and combine various reasoning methods in PSML more efficiently

```

?- find camera megapixel ($ge, 3.5)
camera PowerShot_A95
megapixel=5.0
-----
camera PowerShot_A85
megapixel=4.0
-----
camera DSC-W1
megapixel=5.26
-----
camera DSC-T1
megapixel=5.1

```

Fig. 11. Query-answering in *find* by using both global and local information sources

```
?- find camera megapixel ($ge, 3.5)
camera DSC-W1
megapixel=5.26
-----
camera DSC-T1
megapixel=5.1
```

Fig. 12. Query-answering in *find* by only using local information source

and effectively for complex adaptive, distributed problem solving. Furthermore, distributed Web inference engines need to be implemented on the wisdom Web and knowledge grids (include Web service technology) to deal with huge and multiple distributed information sources [5,6,9,13,14].

Acknowledgement

This work is supported partially by the NSFC major research program: “Basic Theory and Core Techniques of Non-Canonical Knowledge” (60496322), the National Natural Science Foundation of China (60173014), the Natural Science Foundation of Beijing Municipality (4022003), the Natural Science Foundation of Inner Mongolia Autonomous Region (200208020217) and the Open Foundation of Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology. This work is also part of the project supported by a Hong Kong RGC Central Allocation group research grant (HKBU 2/03C).

References

1. Alferes, J. J., Damasio, C.V., Pereira, L. M.: Semantic Web Logic Programming Tools. F. Bry, N. Henze, and J. Maluszynski. (eds.) Principles and Practice of Semantic Web Reasoning, Springer (2003) 16-32
2. Cheung, W., Liu, J., Tsang, K., Wong, H.: Goal-Directed Service Flow Planning Using Metaknowledge on Services, Proc. Second International Workshop on Knowledge Grid and Grid Intelligence, Beijing, China (2004) 123-130
3. Clark, J.: XSL Transformations (XSLT) Version 1.0, <http://www.w3.org/TR/xslt>
4. F. Fencel, F., Motta, E., van Harmelen, F., Benjamins, V. R., Crubezy, M., Decker, S. Gaspari, M., Groenboom, R., Grosso, W., Musen, M., Plaza, E., Schreiber, G., Studer, R., Wielinga, B.: The Unified Problem-Solving Method Development Language UPML, Knowledge and Information Systems 5 (2003) 83-131
5. Foster, I., Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann (2004)
6. Foster, I., Jennings, N. R., Kesselman C.: Brain Meets Brawn: Why Grid and Agents Need Each Other. 3rd International Conference on Autonomous Agents and Multi-Agent Systems, New York (2004) 8-15
7. Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML, Draft Version 0.7 of 30 April 2004, <http://www.daml.org/rules/proposal/>

8. Kang, J., Lee, J. K.: Extraction of Structured Rules from Web pages and Maintenance of Mutual Consistency: XRML Approach. *Ontologies, Rules and Rule Markup Languages for the Semantic Web*, LNAI 2876, Springer (2003) 150-163
9. Liu, J.: Web Intelligence (WI): What Makes *Wisdom Web*? Proc. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03) (2003) 1596-1601
10. Quaresma, P., Rodrigues, I. P.: A Natural Language Interface for Information Retrieval on Semantic Web Documents, E. Menasalvas al. (eds.): *Advances in Web Intelligence*, Springer LNAI 2663 (2003) 142-154
11. Su, Y., Zheng, L., Zhong, N., Liu, C., Liu, J.: Distributed Reasoning Based on Problem Solver Markup Language (PSML) – A Demonstration through Extended OWL –. The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE-05) (2005) 208-213
12. Tomita, K., Zhong, N., Yamauchi, H.: Coupling Global Semantic Web with Local Information Sources for Problem Solving, First International Workshop on Semantic Web Mining and Reasoning, Beijing, China (2004) 66-74
13. Zhong, N.: Developing Intelligent Portals by Using WI Technologies. Li, J. P. et al (Eds.): *Wavelet Analysis and its Applications, and Active Media Technology*, 2, World Scientific (2004) 555-567
14. Zhong, N., Liu, J. (eds.): *Intelligent Technologies for Information Analysis*, Springer Monograph (2004)

An Auto-stopped Hierarchical Clustering Algorithm Integrating Outlier Detection Algorithm

Tian-yang Lv^{1,2}, Tai-xue Su¹, Zheng-xuan Wang¹, and Wan-li Zuo¹

¹ College of Computer Science and Technology, Jilin University, Changchun, China

² College of Computer Science and Technology, Harbin Engineering University, Harbin, China
raynor1979@163.com, wanli@mail.jlu.edu.cn

Abstract. It is a critical problem for the clustering analysis techniques to select the appropriate value of parameters. Meanwhile, the clustering algorithms lack the effective mechanism to detect outliers while treating outliers as “noise”. By regarding outliers as valuable information, the paper proposes a novel hierarchical clustering algorithm that integrates a new outlier-mining method. The algorithm stops clustering according to the dissimilarity reflected by the detected outliers and needs only one parameter, whose appropriate value can be decided in the outlier mining process. After discussing some related topics, the paper adopts 5 real-life datasets to evaluate the performance of the clustering algorithm in outlier mining and clustering and compare it with other algorithms.

Keywords: Clustering; Outlier Mining; Auto Stop.

1 Introduction

Clustering analysis techniques have been widely applied in data mining, pattern recognition, image segmentation [1] and the organization of large databases [2].

It is a critical problem for the clustering analysis techniques to select appropriate value for parameter(s), which is also an important research direction in KDD [3]. For instance, the hierarchical clustering algorithms CURE [4], ROCK [5] and BIRCH [6], and the partitioning algorithm K-means require the user-specified number of final clusters k . However, it is very difficult to choose an appropriate k because of lacking the valuable prior knowledge. Even if such knowledge is available, it maybe inconsistent with data's realistic distribution situation.

Some researches try to make clustering algorithm automatically estimate k . [1] and [2] adopt the global criterion function f : stop clustering once f is optimized. The shortcomings of this method are: prone to fall into local optimization; needing new parameter(s); difficult in choosing the appropriate f . A new clustering method based on dissimilarity increments is proposed in [8]. However, its basic hypothesis that the dissimilarity increment of all data exhibits exponential distribution is not always true for the real-life datasets. The algorithm is also short at handling outlier and detecting clusters with complex shape, such as the linearly inseparable datasets.

The clustering algorithms are also short at detecting outliers. Some have no such ability, while others prune the small clusters or the clusters growing very slowly as

outliers, like CURE, ROCK and DBScan^[7]. The latter scheme actually treats outlier-detection as the byproduct of clustering and can not mine outliers effectively or explain the reason why the abnormality of data occurs. And these algorithms also waste the valuable domain knowledge in deleting the detected outliers as “noise”.

To overcome the above problems, the paper proposes a new strategy that combines outlier detection with the hierarchical clustering process. The strategy intends to stop clustering automatically according to the dissimilarity degree implied by the detected outliers. It is based on the following observation: the distances among data or clusters not only show their similarity degree, but also demonstrate the dissimilarity. And with the progressing of clustering, the dissimilarity $D(C_{NN-A}, C_{NN-B})$ between the two most similar clusters C_{NN-A} and C_{NN-B} at present is increasing. Thus, the clustering should stop at the moment when C_{NN-A} and C_{NN-B} are so diverse from each other. The outlier-mining process can provide the required “diverse degree” since outliers are detected according to their “great difference” from the others.

To take the strategy into realization, the paper proposes a new outlier detection algorithm in section 2. Section 3 states the novel clustering algorithm ASHCA, which is constructed on the traditional hierarchical clustering process and the new outlier mining method. The algorithm ASHCA stops automatically according to outlier information and needs only one parameter, whose value can be decided in the outlier-detection process. After the discussion of some related topics in section 4, section 5 conducts series experiments to compares ASHCA with several other clustering algorithms. And section 6 summarizes the paper.

Table 1. Important Notations

Notation	Description
N	Total number of Data
M	Dimensionality of Data
k	Number of Final Clusters
C_i	The i th Cluster
$D(C_i, C_j)$	Distance between C_i and C_j

2 The Outlier Detection Algorithm Based on the Even-Distribution Pattern

As a separate branch of KDD, outlier detection is to find the data that is considerably dissimilar with the others. The most widely accepted definition of outliers is given by Hawkins^[15]: an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.

And the basic idea of distance-based outlier detection method^[10, 11] is: if the distances between data a and most other data are larger than the threshold D_{out} , a is an outlier. However, it is critical but difficult to decide an appropriate D_{out} and this kind of method ignores the local distribution feature of one data. For instance, in Figure 1,

p is the first outlier candidate because it is the farthest one from the others. But data locating near q are more compressed than those near p , which makes q a more natural outlier.

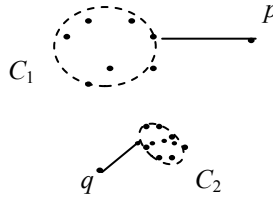


Fig. 1. Local Distribution Situation of Data

This section proposes a new distance-based algorithm, which determines D_{out} according to the even distribution pattern of data and considers the local distribution feature of data in the outlier mining process.

The even distribution pattern is an important reference, since clusters and outliers exist only if the real-life data distribute unevenly. The distances \bar{D}_{NN} between each data and its nearest neighbor are the same if data distribute evenly. To compute \bar{D}_{NN} , the M -dimensional vector space S that N data locate in is equally divided into N grids with only one data in grid's center. Easy to prove that the length of the i th-dimensional edge of a grid is $(a_{max}^{(i)} - a_{min}^{(i)}) / \sqrt[M]{N}$, where $a_{max}^{(i)}$ and $a_{min}^{(i)}$ is the maximum and the minimum of all data's i th-dimension. Obviously, the super-rectangle R decided by a_{max} and a_{min} contains space S . Thus:

$$\bar{D}_{NN} = \sqrt{\sum_{i=1}^M ((a_{max}^{(i)} - a_{min}^{(i)}) / \sqrt[M]{N})^2} \tag{1}$$

To describe the diversity of the realistic distribution situation from the even distribution pattern, parameter β is adopted and $D_{out} = \bar{D}_{NN} / \beta$. Two factors influence the value of β : (1) when data cluster together, the average distance among data or clusters should be larger than \bar{D}_{NN} , which means $D_{out} > \bar{D}_{NN}$; (2) the space R decided by a_{max} and a_{min} is larger than the space S , for example data are approximately distributed on a line, in that case $D_{out} < \bar{D}_{NN}$.

In order to evaluate the local distribution feature of a data, factor ζ is adopted. For data a , $\zeta(a) = D_{NN}(a) / D_{NN}(b)$, where $D_{NN}(a)$ is the distance between a and its nearest neighbor and so is $D_{NN}(b)$. The value of $\zeta(a)$ shows the isolation degree of a from its neighbors. For example, $\zeta(q) > \zeta(p)$ in Figure 2. The special method is adopted to handle the very similar or duplicate data. The reason is that: suppose that b is a 's nearest neighbor and b and c are very similar or the same, which means $D_{NN}(b) \rightarrow 0$, viz. $\zeta(a) \rightarrow \infty$. To avoid this, adjust the equation for $\zeta(a)$ as follows:

$$\xi(a) = \begin{cases} D_{NN}(a) / D_{NN}(b) & \text{if } (D_{NN}(b) > 10^{-4}) \\ 1 & \text{else} \end{cases} \tag{2}$$

Therefore, the outlier evaluation criterion is stated as follows:

$$D_{NN}(a) * \xi(a) > \left(\frac{\sqrt{\sum_{i=1}^M ((a_{max}^{(i)} - a_{min}^{(i)}) / \sqrt{N})^2}}{\beta} \right) \tag{3}$$

Data a is an outlier, if

The following method is proposed to decide the value of parameter β :
 (1) suppose $\beta = \beta_{Step}$ when only one outlier is detected and name β_{Step} as the *step length*, obviously $\beta_{Step} = D_{NN}(a_{forest}) \times \xi(a_{forest}) / \sqrt{D_{NN}}$, where a_{forest} satisfies $D_{NN}(a_{forest}) \times \xi(a_{forest}) \geq D_{NN}(b) \times \xi(b)$ for any b ; (2) observe the increasing speed V of the detected outlier number n_{out} under different value of β , viz. $V = \nabla n_{out} / \nabla \beta = \nabla n_{out} / \beta_{Step}$, where $\beta = l \times \beta_{Step}$ and $l = \{1, 2 \dots\}$, and call l the *step Num.*; (3) if V reaches its first peak when $l_i \times \beta_{Step}$, $\beta = (l_i - 1) \times \beta_{Step}$.

This method is based on following observation: n_{out} increases much faster with further decrease of D_{out} after outliers are detected, since outliers are extremely far away from the others while the normal are relatively near to each other. Taking Figure 2 as an example, after p and q are recognized as outliers, data in cluster C_1 and C_2 will be detected as outliers with the increase of *step num.* l . Figure 2 shows the detail.

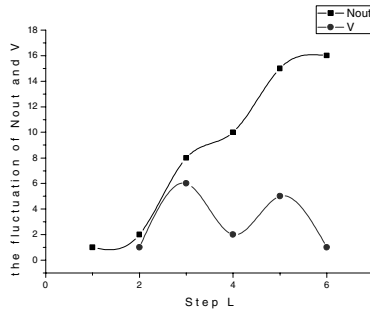


Fig. 2. Example of the Changes of n_{out} and V

Compared with other outlier detection algorithms^[10, 11], the proposed algorithm refers to a more general case, the even distribution pattern, uses only one parameter β and states a semi-automated method to determine β . Obviously, β can be replaced by a pre-defined number of outliers. Since a_{max} and a_{min} can be decided in data reading and β_{Step} is a byproduct in computing the nearest neighbor of all data, these three are the input for the algorithm. The overview of the new algorithm is listed in Fig. 3.


```

Void outlier( $a_{\max}$ ,  $a_{\min}$ ,  $\beta_{\text{Step}}$ )
1. {   decide the value of  $\beta$ ;

      
$$D_{\text{out}} = \sqrt{\sum_{i=1}^M ((a_{\max}^{(i)} - a_{\min}^{(i)}) / \sqrt{N})^2} / \beta$$

2.   ;
3.   for ( $i=0$ ;  $i<N$ ;  $i++$ ) //evaluating the  $i$ th data
4.   { call the nearest neighbor of data  $i$  as data  $j$ 
      ;
5.     if ( $D_{\text{NN}}(j) \geq 10^{-4}$ )
6.        $\xi(i) = D_{\text{NN}}(i) / D_{\text{NN}}(j)$ ;
7.     else  $\xi(i) = 1$ ;
8.     if ( $D_{\text{NN}}(i) * \xi(i) > D_{\text{out}}$ )
9.       data  $i$  is an outlier; }
10.} // End of outlier()

```

Fig. 3. The New Outlier Detection Algorithm

3 The Auto-stopped Hierarchical Clustering Algorithm

The overview of traditional hierarchical clustering process is as follows:

(1) Pre-decide the number of result clusters k and each input data is treated as a cluster; (2) Find the nearest neighbor of each cluster; (3) Merge the closest pair of clusters, if more than k clusters remain, go to (2), else stop. Many ways are available to compute D between two clusters. To be more general, the distance D is defined as the distance between two clusters' center in the following parts.

Compared with the traditional method, the Auto-Stopped Hierarchical Clustering Algorithm ASHCA shows its uniqueness in two aspects, outlier detection and automatic stop.

3.1 Outlier Detection

To compensate the clustering algorithms in outlier mining, ASHCA performs outlier detection in two phases:

- (1) Detecting outliers using the method in section 2 before clustering;
- (2) Treating very small clusters in the clustering result as outliers.

The first phase reduces the interruptions coming from outliers for clustering and the second phase refines the clustering result.

3.2 Automatic Stop

Without user-specified k , it is necessary to extract the stop information from the processed data. As stated in the former parts, it is a suitable opportunity to stop clustering if the clusters to be merged are too dissimilar from each other.

We propose D_{out} as the dissimilarity threshold to decide this opportunity for two reasons: (1) D_{out} is used to detect outliers, while the major characteristic of outliers is their dissimilarity from the others; (2) D_{out} is decided according to the even distribution pattern, which is also a useful reference for clustering since the existence of clusters shows the diversity of the realistic distribution situation from the even pattern.

In practice, the distance between clusters equals the distance of their centers, which means the cluster is represented by its center. Therefore, equation (3) can be adopted as the criterion to decide the auto-stopped opportunity. Since $\zeta(C_{\text{NN-A}}) = \zeta(C_{\text{NN-B}}) = 1$ for the closest clusters $C_{\text{NN-A}}$ and $C_{\text{NN-B}}$, the stop criterion for clustering process is that:

Suppose $C_{\text{NN-A}}$ and $C_{\text{NN-B}}$ are the most similar clusters at present, stop clustering if $D(C_{\text{NN-A}}, C_{\text{NN-B}}) > D_{\text{out}}$.

3.3 Complexity Analysis and Overview of ASHCA

The complexity of traditional hierarchical algorithm is $O(N^2)$. Since ASHCA is constructed on traditional method, it is only necessary to analyze the complexity of each change. The complexity increases by $O(N)$ to perform one more scan to detect outliers, while k is automatically determined and indirectly influences the complexity. Thus, the complexity of ASHCA is $O(N^2)$.

The proposed algorithm ASHCA stop automatically according to the dissimilarity of outlier, thus the parameter k is canceled. Fig. 4 shows its overview.

4 Discussion

As we know, previous researches usually treat clustering and outlier-detection as separate topic and little effort has been taken to combine them.

Meanwhile, the proposed ASHCA algorithm makes clustering work together with outlier detection quite well. First, the outlier-detection algorithm excludes most disturbances of outliers for the afterward clustering; second, the outlier information is utilized in the determination of k ; third, it discovers the common character of outliers by treating the very small final clusters as outliers. And ASHCA achieves all these with less work load, since it determines vectors a_{max} and a_{min} in data inputting and the computing of all data's nearest neighbor is the fore-step for both outlier-detection and clustering.

Table 2 is the theoretical comparison of As-Rock with other clustering algorithms, such as the traditional hierarchical clustering algorithm (TCA), from the following aspects: (1) number of parameters that the respective algorithm needs; (2) whether the suitable value of parameters could be automatically decided; (3) whether the algorithm can stop without user-defined k ; (4) whether the algorithm detects outliers ahead of clustering; (5) its ability to detect clusters with complex-shape; (6) the computational complexity.

```

Algorithm ASHCA( )
1. { while (Not End) //read all M-dimensional data
2. { read data a; // decide vector  $a_{\max}$  and  $a_{\min}$ 
3.   if ( $a^{(i)} > a_{\max}^{(i)}$ )  $a_{\max} = a^{(i)}$  ;
4.   if ( $a^{(i)} < a_{\min}^{(i)}$ )  $a_{\min} = a^{(i)}$  ;
5. }
6. Treat each data as a cluster;
7. for ( $i=0$ ;  $i<N$ ;  $i++$ )
8.   Compute  $C_i$ 's nearest-neighbor;
9. Determine the value of  $\beta_{\text{Step}}$ ;
10. Name the nearest clusters at present as  $C_{\text{NN-A}}$ ,  $C_{\text{NN-B}}$ ;
11.  $D_{\text{out}} = \text{outlier}$  ( $a_{\max}$ ,  $a_{\min}$ ,  $\beta_{\text{Step}}$ ) //detect outliers
12. while ( $D(C_{\text{NN-A}}, C_{\text{NN-B}}) \leq D_{\text{out}}$ )
13. { Merge clusters  $C_{\text{NN-A}}$  and  $C_{\text{NN-B}}$ ;
14.   Update  $C_{\text{NN-A}}$  and  $C_{\text{NN-B}}$ ; }
15. Output the clustering result;
16. } //End of ASHCA
    
```

Fig. 4. The Auto-stopped Hierarchical Clustering Algorithm ASHCA

Table 2. Comparison with Other Clustering Algorithms

Property	ASHCA	TCA	K-Means	CURE	Frozen ^[8]	DBScan
1. Number of Parameters	1	1	1	3	1	2
2. Automatic Retrieval of Parameter Value	Yes	No	No	No	No	No
3. Auto Stopped	Yes	No	No	No	Yes	Yes
4. Mining Outlier before Clustering	Yes	No	No	No	No	No
5. Ability to detect complex-shape clusters	+	+	+	+++	+	++
6. The computation complexity	$O(N^2)$	$O(N^2)$	$O(k*N)$	$O(N^2)$	$O(N^2)$	$O(N^2)$

5 Experiment and Analysis

4 real-life datasets of UCI Machine Learning Repository ^[12] with different kinds of attribute and dimensionality are adopted in the experiment to compare ASHCA with K-Means using random initial points, traditional hierarchical clustering method, DBScan and the algorithm of [8], which is nicknamed as the Frozen algorithm due to its major characteristic. The NBA dataset ^[9] is primarily used to evaluate the performance the outlier detection algorithm of section 2. The details of the 5 datasets are listed in Table 3.

Table 3. Details of the 5 Datasets

Name	N	M	Class Num.	Attribute Type	Missing Attribute Values
NBA	330	3	Unknown	real	No
Zoo	101	16	7	categorical	No
Vote	435	16	2	categorical	Yes
Iris	150	4	3	real	No
Wine	178	13	3	real	No

By applying the FastMap algorithm ^[14], the high-dimensional datasets are mapped to the 2-dimensional vector space in Figure 5. It reflects the inconsistency between the prior knowledge with the realistic distribution situation, since data of different classes locate together and some sub-clusters belong to the same class distribute in the different part.

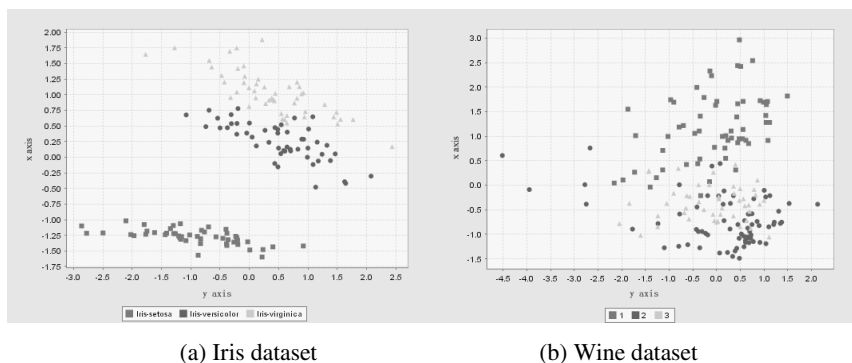


Fig. 5. Mapping the Datasets to 2-dimensional Vector space

To measure the clustering results' quality, two criterions *Entropy* and *Purity* of [13] are adopted:

$$Entropy = \sum_{i=1}^k \frac{n_i}{N} \left(-\frac{1}{\log q} \sum_{j=1}^q \frac{n_i^j}{n_i} \log \frac{n_i^j}{n_i} \right), \quad Purity = \sum_{i=1}^k \frac{1}{N} \max_j (n_i^j) \tag{4}$$

n_i^j is the number of data of j th original class assigned to the i th cluster. The better the clustering result, the smaller is the *Entropy* and the bigger is the *Purity*. For an ideal result, *Entropy*=0.0 and *Purity*=1.0.

5.1 Performance of Outlier Detection

Table 4 gives the details of the detected outliers of the new outlier-detection algorithm. Figure 6 shows the changes of V with the increasing of *step num. l* of the NBA and ZOO dataset. Compared with the outlier-detection algorithm in [11], the new outlier mining algorithm detects not only the out-performing players according to each single attribute and the all-powerful players, but also the worst player Aaron Mckie. And the first detected outlier is Dennis Rodman.

Table 4. The Detected Outliers of NBA Data Set ($\beta_{step}=0.3$ and $\beta=0.9$)

No	Step Num.	Name	<i>Ppg</i>	<i>Rpg</i>	<i>Apg</i>
1	1	Dennis Rodman	4.7	15.0	2.9
2	2	Michael Jordan	28.7	5.8	3.5
3	2	Anthony Mason	12.8	10.2	4.2
4	3	Rod Strickland	17.8	5.3	10.5
5	3	Aaron Mckie	4.1	2.9	2.2.
6	3	Charles Barkley	15.2	11.7	3.2

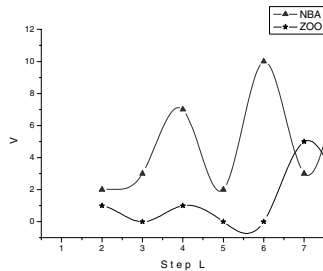


Fig. 6. Changes of V with the Increase of Step L

5.2 Performance of Clustering

The clustering results of ASHCA are listed in Table 5 along with the detected number of outliers. To be more persuasive, Table 5 also gives the best clustering results of DBScan and Frozen under all possible parameters' value and makes the parameter k of K-Means and TCA equal to the number of final clusters obtained by ASHCA. It can be seen that ASHCA performs well for almost each dataset. And the number of the final clusters of ASHCA is more suitable than that of DBScan and Frozen.

Experiment also shows that the proposed value range [3, 5] of parameter α of Frozen is invalid. Actually, it is a hard job to find the appropriate parameters' value for DBScan and Frozen.

Table 5. Overview of the clustering results of ASHCA and other algorithms

	Dataset	Parameters	k	Entropy	Purity	n_{out}
ASHCA	Zoo	$\beta=4.6$	12	0.0137	0.9894	7
	Vote	$\beta=2.4$	2	0.4714	0.8690	9
	Iris	$\beta=2.2$	4	0.2029	0.9034	5
	Wine	$\beta=3.4$	4	0.0275	0.9937	23
Frozen	Zoo	$\alpha=0.7$	10	0.2945	0.7822	--
	Vote	$\alpha=0.7$	21	0.5220	0.8230	--
	Iris	$\alpha=4.0$	2	0.4206	0.6667	--
	Wine	$\alpha=0.5$	13	0.4998	0.7247	--
DBScan	Zoo	$\varepsilon=1.5,MPts=2$	10	0.0127	0.9901	--
	Vote	$\varepsilon=2.5,MPts=3$	2	0.3862	0.5206	--
	Iris	$\varepsilon=0.7,MPts=3$	2	0.4077	0.6867	--
	Wine	$\varepsilon=35,MPts=3$	6	0.5866	0.6798	--
K-Means	Zoo	$k=12$	--	0.1445	0.8713	--
	Vote	$k=2$	--	0.4615	0.8552	--
	Iris	$k=4$	--	0.2227	0.8533	--
	Wine	$k=4$	--	0.7191	0.4874	--
TCA	Zoo	$k=12$	--	0.0520	0.9604	--
	Vote	$k=2$	--	0.9607	0.6161	--
	Iris	$k=4$	--	0.1978	0.9067	--
	Wine	$k=4$	--	0.9561	0.4101	--

6 Conclusion

To overcome the shortcomings of traditional approaches, the paper states a new strategy that integrates outlier detection with clustering and proposes an auto-stopped hierarchical clustering algorithm ASHCA, which is constructed on a new outlier-detection method and needs only one parameter. Experimental results show ASHCA's good performance in both outlier detection and clustering.

The future works will concentrate on the study of ASHCA to handle large dataset. Although it is feasible to cluster the sample of large dataset, it is still an open question to perform clustering and outlier detection on the rest of the dataset during the scanning process, especially when the sample set can not represent the actual distribution situation of the whole set.

Acknowledgements

This work is sponsored by the Natural Science Foundation of China under grant number 60373099 and the Natural Science Research Foundation of Harbin Engineering University under the grant number HEUFT05007.

References

1. C. Rosenberger, K. Chehdi: Unsupervised Clustering Method with Optimal Estimation of the Number of Clusters: Application to Image Segmentation. *International Conference on Pattern Recognition*, Volume 1 (Sep. 2000) 656-659
2. Xuejian Xiong. Kap Luk Chan. Towards: An Unsupervised Optimal Fuzzy Clustering algorithm for Image Database Organization. *International Conference on Pattern Recognition*, Volume 3. (Sep. 2000) 3909-3913
3. Johannes Gehrke: Report on the SIGKDD 2001 Conference Panel "New Research directions in KDD", *SIGKDD Explorations*, 3(2), (2002) 76-77
4. S. Guha, R. Rastogi, K. Shim: CURE: an Efficient Clustering Algorithm for Large Database. In: Laura M. Haas and Ashutosh Tiwary, eds. *Proceedings of the ACM SIGMOD Conference on Management of Data*. Seattle, Washington: ACM Press (1998) 73-84
5. S. Guha, R. Rastogi, K. Shim: ROCK: a robust clustering algorithm for categorical attributes. In: *Proc. of the 15th Int'l Conf. on Data Eng (1999)* 512-521
6. Tian Zhang, etc: BIRCH: An Efficient Data Clustering Method for Very Large Databases. In: *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (1996)* 103-114
7. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, OR (1996) 226-231
8. Ana L.N. Fred, José M.N. Leitão: A new Cluster Isolation criterion Based on Dissimilarity Increments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, VOL. 25, No. 8 (August 2003) 944-958
9. <http://www.statdaddy.com/>
10. Edwin M. Knorr, Raymond T. Ng: Finding Intensional Knowledge of Distance-Based outliers. In: *Proceedings of the 25th Very Large Data Bases conference*. Edinburgh, Scotland (1999) 211 - 222
11. Sridhar Ramaswamy, Rajeev Rastogi, Kyuseok Shim. Efficient Algorithms for mining outliers from Large Data Sets. *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. Dallas, Texas, United States (2000) 427 – 438
12. Hettich, S. & Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science.
13. Ying Zhao, George Karypis: Criterion Functions for Document Clustering: Experiment and Analysis. Technical Report #01-40, University of Minnesota (2001) 1 – 40
14. C. Faloutsos, K. Lin: FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In: *Proceedings of 1995 ACM SIGMOD, SIGMOD RECORD*, vol.24, no.2 (June 1995) 163-174
15. Hawkins D: Identification of Outliers. Chapman and Hall, London. (1980)

Research Paper Recommender Systems: A Subspace Clustering Approach*

Nitin Agarwal, Ehtesham Haque, Huan Liu, and Lance Parsons

Arizona State University, Tempe AZ 85281, USA

Abstract. Researchers from the same lab often spend a considerable amount of time searching for published articles relevant to their current project. Despite having similar interests, they conduct independent, time consuming searches. While they may share the results afterwards, they are unable to leverage previous search results during the search process. We propose a research paper recommender system that avoids such time consuming searches by augmenting existing search engines with recommendations based on previous searches performed by others in the lab. Most existing recommender systems were developed for commercial domains with millions of users. The research paper domain has relatively few users compared to the large number of online research papers. The two major challenges with this type of data are the large number of dimensions and the sparseness of the data. The novel contribution of the paper is a scalable subspace clustering algorithm (SCuBA¹) that tackles these problems. Both synthetic and benchmark datasets are used to evaluate the clustering algorithm and to demonstrate that it performs better than the traditional collaborative filtering approaches when recommending research papers.

1 Introduction

The explosive growth of the world-wide web and the emerging popularity of e-commerce has caused the collection of data to outpace the analysis necessary to extract useful information. Recommender systems were developed to help close the gap between information collection and analysis by filtering all of the available information to present what is most valuable to the user [20].

One area of the web that has seen continued growth is the online publication of research papers. The number of research papers published continues to increase, and new technology has allowed many older papers to be rapidly digitized. A typical researcher must sift through a large quantity of articles manually, relying on keyword-based searches or paper citations to guide them. The search results of researchers with similar interests can help direct a more effective search, but the process of sharing search results is often too cumbersome and time consuming to be feasible. A recommender system can help by automatically recommending papers based on the preferences of other researchers with similar interests.

* Supported by grants from Prop 301 (No. ECR A601) and CEINT 2004.

¹ SCuBA: Subspace Clustering Based Analysis.

There are two main branches of recommender systems; content based filtering and collaborative filtering. Content based filtering (CBF) approaches create relationships between items by analyzing inherent characteristics of the items. Collaborative filtering (CF) systems do not analyze an items properties, but instead take advantage of information about users' habits to recommend potentially interesting items. The analysis of user behavior patterns, allows collaborative filtering systems to consider characteristics that would be very difficult for content based systems to determine such as the reputation of the author, conference, or journal. CF approaches are also well suited to handle *semantic heterogeneity*, when different research fields use the same word to mean different things.

The remainder of the paper is organized as follows. In Section 2 we discuss the common challenges in the collaborative filtering process followed by a formal definition of the problem we propose to solve. In Section 3 we give a detailed description of our proposed algorithm. In Section 4 experimental results are presented, including description of the dataset used, the evaluation metrics and discussion. In Section 5 we discuss related work in the area of subspace clustering and recommender systems. Finally Section 6 contains concluding remarks and directions for future research.

2 Challenges, Definitions and Problem Statement

In spite of the success achieved by CF algorithms, there are limitations to such systems [10] that arise in the research paper domain. In many domains, there is an ever increasing number of users while number of items remains relatively stable. However, in research paper recommendation domain, the number of users (researchers) is much less than the number of items (articles). Collaborative filtering systems face two major challenges in the research paper domain: scalability to high dimensional data and data sparsity. In a typical recommender system there are many items. For example, Amazon.com recommends specific books of interest from a large library of available books. Item-based approaches that determine similarity measures between items do not perform well since the item space is extremely large. A user based approach allows us to leverage the relatively small number of users to create an efficient algorithm that scales well with the huge number of research papers published each year. An intuitive solution used by early collaborative filtering algorithms is to find users with similar preferences to the current user and recommend other items that group of users rated highly. Even with a relatively small number of users, however, this approach is computationally complex. The use of clustering algorithms to pre-determine groups of similar users has been used to significantly increase performance [16].

Presence of sparsity poses a problem for user-based approaches because they often rely on nearest neighbor schemes to map a new user to the existing user groups. It has been demonstrated that the accuracy of nearest neighbor algorithms is very poor for sparse data [4]. Subspace clustering is a branch of

clustering algorithm that is able to find low dimensional clusters in very high-dimensional datasets. This approach to clustering allows our system to find groups of users who share a common interest in a particular field or sub-field regardless of differences in other fields. Searching for similar users across all of the items leads to finding users who share many common interests. We address the issue of high-dimensionality and sparsity of the data space by proposing a new approach to collaborative filtering utilizing subspace clustering principles. Furthermore, we propose a novel subspace clustering algorithm suited to sparse, binary data.

We define the data space in the research paper domain as an $m \times n$ matrix such that there are m researchers $R = \{ r_1, r_2, \dots, r_m \}$ and n articles $A = \{ a_1, a_2, \dots, a_n \}$. The row r_i represents the interests of researcher i and consists of a list of articles A_{r_i} which indicates the user's interest in those articles. In the research paper domain, this could indicate that the user has read or accessed a certain article. For a given session there is an *active researcher* ($r_{active} \in R$) for which the collaborative filtering algorithm would like to recommend new articles that may be of interest to r_{active} based on the researcher's interest in the current session and the opinion of other like-minded researchers.

In order to predict which articles will be of high interest, we must have models of like-minded researchers. Given the $m \times n$ matrix, we can find like-minded researchers by finding groups of researchers $r \subseteq R$ who have expressed interest in similar articles $a \subseteq A$. The problem of finding such groups can be transformed into a problem of *subspace clustering* using the previously described binary matrix as input. The result of subspace clustering would be clusters, of researchers in corresponding subspaces of articles. Here, the underlying assumption is if a group of researchers have similar interests then they usually access similar sets of articles or vice-versa.

Problem Statement. Given a binary $m \times n$ matrix, where rows represent m researchers ($R = \{ r_1, r_2, \dots, r_m \}$) and columns represent n articles ($A = \{ a_1, a_2, \dots, a_n \}$), find subspace clusters of researchers, $r \subseteq R$, defined in subspaces of articles, $a \subseteq A$.

3 Proposed Algorithm

The access patterns of the researchers can be maintained by tracking the log of research papers they access. From these access patterns, we can generate a researcher/article data space as defined in the previous section. We infer that people accessing a similar set of articles are interested in the topics represented by the articles. Such a group is represented by a subspace cluster in the researcher/article data space. Finding these experts will ultimately help us achieve our goal of finding the groups of articles that form fields of interest.

Our proposed subspace clustering algorithm is a two-step process starting with finding subspaces that form clusters, then post-processing to remove redundancy. The output of these steps are sub-matrices of the original data matrix. Before

discussing the above steps in detail, we will show how the proposed algorithm addresses the challenges associated with sparse, high-dimensional, binary-valued data in the research paper domain.

3.1 Challenges of the Domain - Addressed

Sparsity and Binary Data. High sparsity means that for a given r_i , which is a vector in n dimensional space, most of the entries in the vector are zeros. Since we are only interested in the values which are not zeroes, the original vector is transformed into a string containing positions of non-zero values. An example is shown in Figure-1. The result of the transformation is compact representation resulting in reduced memory requirements and less processing overhead.

High-Dimensional Data. In high dimensional data, the number of possible subspaces is huge. For example, if there are N dimensions in the data, the number of possible subspaces is 2^N . Hence, subspace clustering algorithms must devise an efficient subspace search strategy [17]. Most existing algorithms will work well when the number of dimensions is relatively small. In the research paper domain there are thousands of dimensions and such approaches may become impractical. For example, based on our work in [17], we chose an efficient representative subspace clustering algorithm, MAFIA [7], to run on a data set with 1000 dimensions. The program ran out of memory because the algorithm was designed to be very efficient for datasets with many rows but comparatively few dimensions. In the research paper domain, we have a unique property that the number of rows is significantly less than the number of dimensions. We overcome this challenge of high-dimensional data by devising an algorithm that exploits the row-enumeration space rather than the larger dimension space.

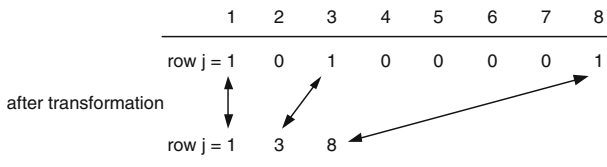


Fig. 1. Transformation: A row in the original matrix is transformed into a string containing the position of the non-zero columns

By combining our exploitation of the row-enumeration space and compact representation of binary sparse data, we convert the challenges into advantages in our subspace clustering algorithm. The next two sections discuss the major steps of our algorithm in more detail.

3.2 Find Subspaces That Form Clusters

Most straightforward way to find subspaces is to project the instance in all possible subspaces and find the ones in which clusters are formed. Such an approach

is not feasible for data with a large number of dimensions since number of possible subspaces is exponential, $O(2^N)$. Therefore, we propose a search strategy which explores the smaller row-enumeration space.

Subspace Search: The result of the transformation (as shown in Figure-1) of the high-dimensional data is a list of strings representing rows in the original data. We call the new, transformed dataset D . An example is shown in Figure-2.

	1	1	2	3	17	
row id	2	1	2	3		
	3	2	3	4	50	
	4	1	2	3		
	5	2	3	4		
	6	2	3			
	7	5	6	7	8	21
	8	6	7	40		
	9	5	6	7	8	26

Fig. 2. Transformed Data D after being compressed

The subspace search proceeds by comparing row_i with each successive row ($row_{i+1}, row_{i+2}, row_{i+3}, \dots, row_m$). For example, if we start at row_1 in Figure-2, we first find the intersection between row_1 and row_2 . The result of the intersection is $1\ 2\ 3$ which represents a subspace in dimension $1, 2$ and 3 with row_1 and row_2 as cluster members. $1\ 2\ 3$ is stored as a key in a temporary hash table and the number of cluster members is stored as the value in the table. In addition, we also store the row-id as the values in the table in order to keep track of the cluster members for a given subspace. Next, row_1 is compared with row_3 and the intersection $2\ 3$ is placed in the hash table. The intersection of row_1 and row_4 , $1\ 2\ 3$, is already present in the table, so the count value is updated. At the end of the pass for row_1 , the hash table will have two entries $1\ 2\ 3$ and $2\ 3$. At this point, the entries in the temporary hash table are put in a global hash table which only accepts entries which are not already present in the global table and the temporary hash table is flushed. The rationale for having this rule is the following. When the search commences at row_2 , it will find the intersection $1\ 2\ 3$ and eventually it will update the global hash table with its local one. Notice here that the subspace $1\ 2\ 3$ has already been found during the search commencing from row_1 . Therefore, there is no requirement to update the global table. At the end of the search, the global hash table will have five entries, $5\ 6\ 7\ 8, 1\ 2\ 3, 2\ 3, 2\ 3\ 4$ and $6\ 7$. Notice that the subspace $2\ 3$ is subsumed by the subspace $1\ 2\ 3$ or $2\ 3\ 4$. This redundant subspace is removed in the next step. The formal description of the algorithm is shown in Figure-3.

Memory Usage: The main sources of memory consumption are the temporary and global hash tables, and the transformed dataset. The hash table memory requirement grows slowly since the temporary hash table is flushed every time a new search commences and the global hash table only contains previously un-found entries. Although use of a hash table may lead to some overhead of space

due to unmapped entries, the advantage of constant time lookup greatly outweighs such overhead. Generally, it was noticed that the memory requirements grew linearly and were stable during our experiments.

Time Complexity: The algorithm takes advantage of the fact that the number of rows, m , is relatively small. As a result, the subspace search is performed on the row enumeration space which is $O(m^2)$. It should be noted that in our case, it is actually less than m^2 because if we are at row_i , we only look at $row_{i+1}, row_{i+2}, \dots, row_m$. The algorithm also requires finding intersection of two strings which is performed in $O(k)$ time where the k is the length of the strings. Notice that k is usually very small due to the high sparsity of the data. In summary, the total complexity is $O(m^2k)$.

Input: Transformed data D with number of rows m and **minimum density** count.
Output: A set S of subspaces.

```

hash_table_temp;
hash_table_global;

for j = 0; j < m; j++
  get row-j;
  for k=j+1; k < m; k++
    get row-k;
    find_intersection(row-j, row-k);
    put intersection in hash_table_temp;
    update count in hash_table_temp;

  if entries of hash_table_temp not in hash_table_global
    put in hash_table_global;

for j = 0; j < hash_table_global.size(); j++
  if count of an entry e >= minimum density
    S += e;

```

End

Fig. 3. Subspace Clustering Algorithm

3.3 Post-processing to Remove Redundancy

A larger subspace which contains several smaller subspaces covers more articles more articles within the same field of interest. Removing smaller subspaces subsumed by larger ones helps in making recommendation process faster in the absence of redundant subspace clusters.

The result from the previous step is a collection of subspaces, S . An example of such a collection is shown in Figure-4. The subspaces connected with arrows indicate two subspaces, one of which subsumes another. We must remove the fourth subspace which is $6 \ 7$ since it is subsumed by subspace $5 \ 6 \ 7 \ 8$. In general, to remove the redundant/subsumed subspaces in S , we perform the following steps:

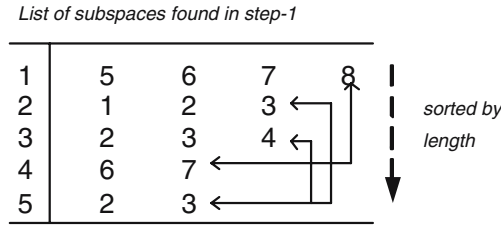


Fig. 4. Set of Subspaces S with redundancy. Subspaces marked with arrows are pairs where one subspace is subsumed by another. For example, subspace 4 is subsumed by subspace 1.

1. Sort the set S according to the size of each subspace in descending order. The result of sorting is shown in Figure-4.
2. Take an element s_i from the set, S , and pass through $s_{i+1}, s_{i+2}, \dots, s_{|S|}$ removing any element if it is a subset of s_i .

By performing step one, we place the largest subspace as the first element in S . Then performing step 2, starting with the first element of the set, we will remove all subsets of s_1 , in the first pass. Since s_1 is the largest subspace, without loss of generality, we can assume that there will be a large number of subsets of s_1 in S . As a result, $|S|$ will shrink considerably and the remaining passes through S will be shorter, resulting in reduced running time.

The time complexity is $O(|S|^2 p)$ where p is the size of the subspaces when computing subsumption between two subspaces. Notice that p is quite small due to sparsity and $|S|$ is shrinking with each iteration. The time complexity for sorting is $O(|S| \lg |S|)$, so the overall complexity is still $O(|S|^2 p)$.

3.4 Finding Overlapping Subspaces

In real world data or data which is highly sparse, it might be possible that subspace clusters in the form of sub-matrices will not be significant in number and size. In that case, we relax our subspace search so that we can find clusters of irregular shape as opposed to strict sub-matrices. These irregularly shaped clusters are larger in size and cover more of the data. Overlapping of these subspace clusters can represent extended or implicit relationships between both items and users which might be more interesting to the user.

An example is shown in Figure-5. Four subspace clusters are grouped together because they share a number of common subspaces. We apply a simple clustering algorithm to cluster the subspaces. For each element in the list of subspaces, the overlap between the given element and the other subspace clusters are found. The degree of overlap is controlled by threshold parameter indicating the percentage of dimensions that match. If the overlap is above the given threshold, the original subspace cluster is selected as a member of the cluster and is considered the *seed* of the cluster. For example in Figure-5, the seed of the new cluster found is the subspace $A B C D$. The other members of this cluster have some degree of overlap

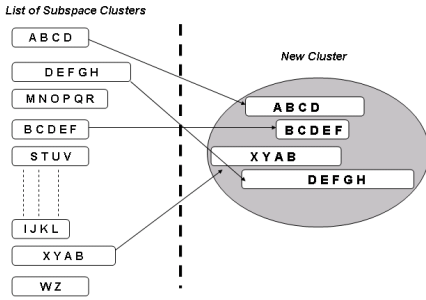


Fig. 5. The subspaces are grouped together in one cluster if there is overlap between subspaces

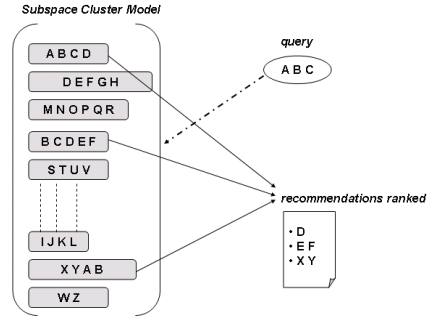


Fig. 6. Using the subspace clustering model to generate recommendations

with $A B C D$. In this example, subspaces with at least one item in common with the seed, are put into the new cluster. This process is repeated for each element from the list of subspaces, resulting in large clusters of subspaces. The user can control the threshold parameter depending on the domain and the data characteristics. A feature of this clustering process is that we allow a subspace to be member of several clusters as opposed to forming discrete partitions. This allows us to find all of the relationships that a subspace may participate in, instead of restricting it to one cluster membership.

3.5 Generating Recommendations

The process of generating recommendations illustrated in Figure-6 involves mapping a user’s query to the subspaces and making recommendations based on the subspaces. The query represents the current selection of the active user. All of the subspaces containing the query item are collected and the matching subspaces are ranked based on the *coverage* of the query. Coverage is defined as the number of query items present in the subspace. The subspace with the highest coverage is ranked first and the ranked order of the subspaces determines the ranking of the recommendations. The recommendations are the elements in the subspace that are not part of the query. For example, in Figure-6 for the query $A B C$, the subspace $A B C D$ has the highest coverage so the recommendation from that subspace, D , is ranked first. In the case of a tie while ranking, the subspace with the higher *strength* is picked first where the strength is defined as the number of cluster members present in the subspace. In Figure-6, subspaces $B C D E F$ and $X Y A B$ have equal coverage. In this case, their ranking is determined by their cluster strengths.

3.6 Fall-Back Model

Since the process of generating recommendations is dependent upon the successful mapping of a query to the subspace clustering model, we consider the scenario

where a query is not covered by the subspace clustering model. Although this is a rare case (see Section 4.4), there must be a mechanism to handle such a scenario if it arises.

The fall-back model utilizes the researcher/article matrix directly. For a given query, the articles in the query are indexed in the matrix and the corresponding rows (researchers) are found. The articles in the rows are ranked according to their global frequency and the recommendations are made in the ranked order. This approach is similar to the user-based approach where the items in the nearest user-vector are recommended to the active user. Notice that the computational requirements of the fall-back approach are minimal, consisting mainly of indexing the query articles and ranking according to the article frequency. Article frequency information can be maintained in a table enabling inexpensive look up cost.

Finally, recommender systems generally work under the principle that a queried item exists in the user/item matrix. An interesting scenario is, when a researcher selects an article that does not exist in the researcher/article matrix. The consequence will be that both the fall-back model and the subspace clustering model will not be able to cover the query. In this case, the top-N frequent (or popular) articles are returned to the researcher. Here, the quality of the recommendation will be poorer than the fall-back model but the goal of covering the query will be satisfied.

4 Experiments

We first evaluate our subspace clustering algorithm using synthetic data and then evaluate the recommendation approach with the benchmark data. With synthetic data, we can embed clusters in specified subspaces. Since we know these locations we can check whether the clustering algorithm is able to recover the clusters. For evaluating the quality of recommendations, *MovieLens* [10] benchmark dataset has been used. We compare our approach using SCuBA with the baseline approach defined in Section 4.2. We present the experimental details in the following subsections.

4.1 Clustering Evaluation on Synthetic Data

We have developed a synthetic data generator that allows us to embed clusters in subspaces. We can control the number and the size of such clusters and also size of the dataset. Apart from the clusters, the data space is filled with noise. For each row we place x noise values at random positions where $x = \alpha \times$ the number of dimensions. We set $\alpha = 1\%$. An example of a synthetic data with three clusters is shown in Figure-7.

For scalability, we measure the running time as we increase the number of dimensions in increments of 1000. The number of rows is kept fixed at 1000. We embed 5 subspace clusters. Each subspace cluster has 20 dimensions and 10 instances. Figure-8 shows the scalability of our algorithm as we increase the dimensions in the data. Notice that the curve is linear since our subspace search

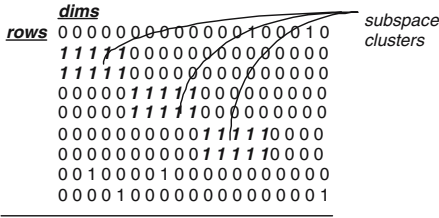


Fig. 7. Example of synthetic data used to evaluate the subspace clustering algorithm. In this example, 3 subspace clusters are embedded and noise is placed at random positions.

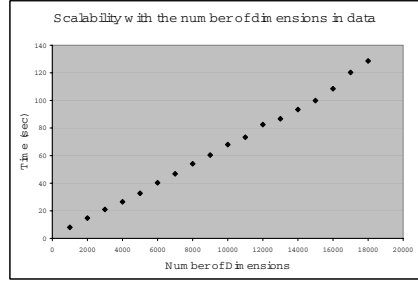


Fig. 8. Scalability with the number of dimensions in the data set

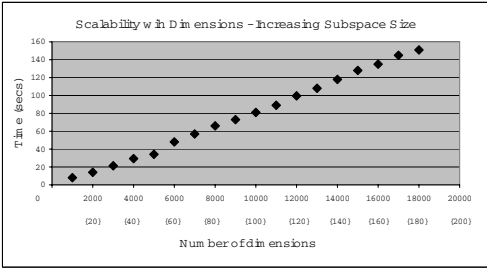


Fig. 9. Scalability with dimensionality of the embedded clusters. Size of the subspace (number of dimensions) is increased linearly with the number of dimensions in the data set. Subspace size, indicated in curly braces, is set to be 1% of dimension size.

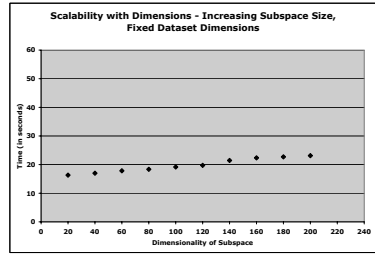


Fig. 10. Scalability with dimensionality of the embedded clusters. Size of the subspace (number of dimensions) is increased linearly keeping the number of dimensions in the data set fixed.

strategy is not dependent on the number of dimensions but rather it searches in the row-enumeration space.

In second scalability graph, shown in Figure-9, we linearly increase the size of subspaces with the number of dimensions in the data. In this case, we set the size of the subspace to be 1 percent of the number of dimensions. For example, when the number of dimensions is 5000, size of the subspace is 50. Here the running time is negligibly higher than the previous case but the curve is still linear. Higher running time is due to the computation of intersection, k , between two strings to check for redundancy as mentioned in Section 3.3. As discussed previously, the size of k is generally very small due to high sparsity of data.

In the third scalability experiment as shown in Figure-10, we check the behavior of SCuBA as the density of the dataset increases. Dimensionality of the dataset is fixed to 2000 and the number of instances to 1000. We embed 5 subspace clusters each with 10 instances. The subspace size is increased from 20

Table 1. Accuracy in recovering the embedded subspace clusters. Five datasets with increasing number of dimensions. In all cases, the 5 embedded clusters are recovered.

Data Dimensions	Recovery Accuracy
1000	5/5 100 %
2000	5/5 100 %
3000	5/5 100 %
4000	5/5 100 %
5000	5/5 100 %

to 200 in steps of 20. It can be observed that running time increases with the density of the dataset but the curve remains linear.

We present accuracy results of our subspace clustering algorithm in Table-1. Here, accuracy is defined by the number of true positives and true negatives w.r.t. the recovered clusters. In all cases, the embedded clusters were completely recovered as shown in Table-1. No extra clusters were reported even though $\alpha = 1\%$ of noise is present in the data.

4.2 Recommendations from Benchmark Data

Here we evaluate the quality of the recommendations made by the SCuBA approach on the *MovieLens* dataset. In Section 1, we reviewed two approaches in CF and pointed out that memory-based approach produce high quality recommendations by finding the nearest neighbors of a target user. We use this as our baseline approach. It is quite practical to assume that users view or rate very few articles of the thousands of available. Since we want to make recommendations based on the few articles a user looks at, we show that when the number of selected terms are few, our approach produces higher quality recommendations than the baseline approach.

Precision and *recall* are widely used measures to evaluate the quality of information retrieval systems. In our experiments, we define quality using precision which is the ratio between number of relevant results returned and the total number of returned results. We choose this measure for the following reasons. The goal of a recommender system is to present a small amount of relevant information from a vast source of information. Therefore, it is more important to return a small number of recommendations that contains relevant items rather than giving the user a large number of recommendations that may contain more relevant recommendations but also requires the user to sift through many irrelevant results. The ratio between the number of relevant results returned and the number of true relevant results is defined as recall. Notice it is possible to have very high recall by making a lot of recommendations. In the research paper recommendation domain, a user will be more interested in reading papers that really qualify for his interests rather than going through a huge list of recommended papers and then selecting those which are of interest. Precision more accurately measures our ability to reach our goal than recall.

Experimental Setup: We divide the data into training and testing sets with 80% used for training and 20% for testing. For our subspace clustering approach, we build subspace clustering models from the training data and for the baseline approach we use the training data to find similar users and make recommendations based on those similar users. During the testing phase, for each user from the test set we take a certain percentage of the items from the test row. We call these *query items*. The rest of the items in the test row are called *hidden items*. We make recommendations (using both approaches) for the user, based on the query items. The list of recommended items are compared with the hidden items and the intersection gives the number of relevant recommendations (results) returned. This forms the basis of our precision measure.

Results and Discussion: The precision curve in Figure-11 shows that we perform better than the baseline approach as we reduce the percentage of query items. As the query items decrease, both relevant recommendations and total recommendations also decrease. In SCuBA, the decrease in relevant recommendations is less than the decrease in total recommendations which is not the case with the baseline approach. Therefore an increase in the precision value is observed. The results validates the discussion presented in Section 1 where it was pointed out that although the user comparison approach produces very high quality recommendations, it will not perform well in our domain where we would like to make recommendations based on very few query terms. Moreover, user comparison approach does not scale linearly with the number of dimensions as shown in Figure-14. These results also verify the fact that more focussed relations are captured using our approach.

The user comparison approach treats users as vectors and computes the similarity between two vectors. The similarity calculations will be poor when there are only a few terms in the test row. In other words, this approach requires large user profiles (similar to e-commerce applications) to generate high quality recommendations which in turn warrants user-tracking and raises privacy issues. In our case we do not require user-profiles that saves the overhead of user-tracking and preserves privacy as well. At a given instant, a researcher may be interested in a new research topic(s), and if we use the researcher’s previous profile

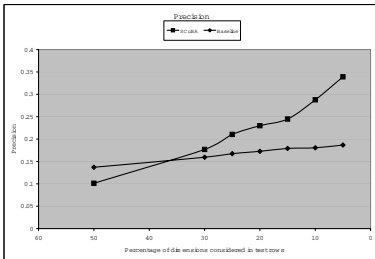


Fig. 11. Precision measurements as the percentage of query items is reduced

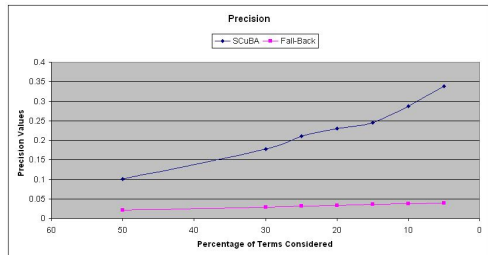


Fig. 12. Comparison of Precision values between SCuBA and Fall-Back model

or likings, we will not find relevant articles matching his/her current interest(s). With SCuBA approach we can overcome this challenge as shown in the precision curve.

4.3 Model Building Times

In model-based approaches a model is created using item similarity [5]. Since, the complexity of building the similarity table is dependent on the number of items, this approach would be unnecessarily computationally expensive in the research paper domain where we have large number of articles but much smaller number of users. Our proposed solution takes advantage of the small number of users and avoids dependence on the number of items. Hence, we would expect that the time required to build models following the subspace clustering approach would be much less than the above approach in [5].

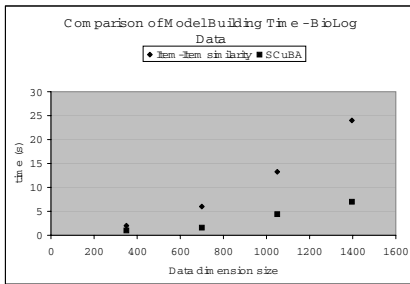


Fig. 13. Time comparison of building models with two approaches on Biolog Dataset

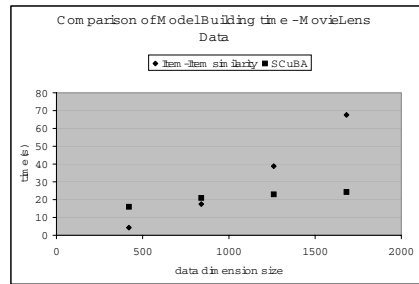


Fig. 14. Time comparison of building models with two approaches on MovieLens Dataset

Our claim is validated by the results shown in Figure-13 and Figure-14. Here, we measure the time taken to build models from the two data sets used in the experiments. The subspace clustering approach clearly outperforms the item-item similarity approach.

4.4 Coverage Results

Here we present statistics on query coverage of the subspace clustering model. As was discussed earlier, we anticipated the subspace clustering model will not be able to provide complete coverage of queries and hence we proposed a fall-back model. Results shown in Table-2 validate our hypothesis but more importantly they show that percentage of queries not covered by subspace clustering model is very low. This means fall-back model is rarely used and hence the overall quality of recommendation will not suffer too much.

Table 2. Percentage of queries that were not covered by the subspace cluster model

Query Length	1	2	3	4	5
Miss Percentage %	8.5	0.5	0.5	0.5	0

Experiment was performed with the *MovieLens* dataset. Here the query length denotes the number of terms considered in the test row. The total number of test rows considered is 188 which is 20% of the complete dataset, 80% of which was used to construct the subspace cluster model. For queries of length 1, there were 16 test rows out of 188 for which no recommendation was made, or the miss percentage is 8.5%. For query lengths greater than 4, miss percentage become zero or some recommendation was made.

4.5 Comparing Subspace Cluster Model with Fall-Back Model

In Section 3.6, fall-back model was introduced which is used when there is no mapping of query terms to the subspace cluster model. In these experiments we try to show that the fall-back model is not a replacement for subspace clustering model. The fall-back model helps to avoid those situations when subspace cluster model fails to make recommendations and it should be used for that purpose only. The experimental setup is kept the same as in Section 4.2. Again, we compare *Precision* for both the models, SCuBA as well as the fall-back. The precision curve in Figure-12 clearly shows SCuBA performs far better than the fall-back model for different values of query terms considered. These results also support the discussion in Section 3.6, the goal of the fall-back model is to provide coverage for query items even if the quality of recommendations is compromised.

5 Related Work

Research paper recommendation systems are mainly content based systems, utilizing a combination of text based analysis as well as the citations of each paper to generate recommendations. Collaborative filtering algorithms have been very successful in other domains, however, and their application to research paper recommendation has not been fully explored. Most collaborative filtering systems generate models in order to scale up massive numbers of users and items. We developed SCuBA, based on the principles of subspace clustering, to generate the collaborative filtering models to recommend research papers. This section explores related work in both recommender systems and subspace clustering.

5.1 Recommender Systems

Content Based recommender systems attempt to determine relationships between items by comparing inherent characteristics of items. In research paper domain, the *CiteSeer* system utilizes the citation network between papers to find related papers [8]. *CiteSeer* also utilizes text-based analysis, but as a separate list of recommendations. *Quickstep* and *FoxTrot* both utilize ontologies of research topics to assist in recommendation [15]. McNeer et. al. propose a method to combined the citation network with various existing CF algorithms [14].

Collaborative filtering approaches can be divided into user-based and model-based approaches. The nearest neighbor, user-based approaches make recommendations by examine the preferences of similar users. Model-based approaches attempt to improve performance by building models of user and item relationships

and using those models to make recommendations. Early CF systems compare the active user to all of the other users and found the k most similar users [22]. Weights are then assigned to items based on the preferences of the neighborhood of k users, using a cosine or correlation function to determine the similarity of users. Recommendations are made using the weighted list of items. The recommendations produced are high quality and the systems are easily able to incorporate new or updated information, but the approaches do not scale well.

To overcome the scalability issues, model based systems were developed. These systems pre-build a user or item based model that is used to make recommendations for an active user. There are two major categories of models, user based and item based. Aggarwal et. al. introduced a graph-based approach where the nodes were users and the edges their similarity. Bayesian probability networks also proved to be useful in building models [2]. Performance was further improved by using dependency networks [9]. Using a bipartite graph, Huang et. al. were able to find transitive associations and alleviate the sparsity problem found in many recommender system datasets [12]. Hofmann was able to discover user communities and prototypical interest profiles using latent semantic analysis to create compact and accurate reduced-dimensionality model of a community preference space [11]. Sarwar et. al. used correlations between items to build models [21]. Recently, Demiriz borrows from clustering approaches and uses a similarity measure to find rules, instead of an exact match [4].

5.2 Subspace Clustering Algorithms

Subspace clustering algorithms can be broadly categorized based on their search method, top-down or bottom-up [17]. Top down approaches search in the full dimensional space and refine the search through multiple iterations. Searching in all of the dimensions first means they are not well suited for sparse data such as that found with recommender systems. Bottom-up approaches first search for interesting areas in one dimension and build subspaces. This approach is much more suited to sparse datasets where clusters are likely to be found using fewer than 1% of the dimensions.

CLIQUE was the first bottom-up algorithm and follows the basic approach [1]. Adaptations to the basic method include *ENCLUS* which uses entropy instead of measuring density directly [3] and *MAFIA* which uses a data-driven adaptive method to form bins [7]. *CLTree* uses a decision tree algorithm to determine the boundaries of the bins [13]. Each of these algorithms focus on continuous valued data and do not perform well on categorical or binary data. Recently there have been subspace clustering algorithms developed for binary [18] and categorical data [19]. The few algorithms designed for both sparse and binary high-dimensional data do not cluster in subspaces of the dataset [6].

6 Conclusions and Future Work

In this paper, we proposed a subspace clustering approach for recommender systems aimed at the research paper domain. A useful source of information when

recommending research papers is the reading habits of other researchers who are interested in similar concepts. Thus, we adopted a collaborative filtering approach which allows us to use data collected from other researchers browsing patterns, and avoids issues with the interpretation of content. Such data consists of a small number of users (researchers) and a very large number of items (research papers). Our proposed approach takes advantage of the unique characteristics of the data in this domain and provides a solution which is fast, scalable and produces high quality recommendations.

In order to improve the perceived quality and usefulness of the recommendations, a more sophisticated ranking scheme could be developed as an extension to the algorithm. The algorithm could also be extended to include the subjective user ratings rather than treating them as binary values and categorize recommendations as strong, mediocre and weak. A lot of work has been done in mixing the two models, content based filtering and collaborative filtering, to generate a hybrid model which tries to enhance the recommendation quality.

References

1. Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 94–105. ACM Press, 1998.
2. John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence.*, 1998.
3. Chun-Hung Cheng, Ada Waichee Fu, and Yi Zhang. Entropy-based subspace clustering for mining numerical data. In *Proceedings of the fifth ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 84–93. ACM Press, 1999.
4. Ayhan Demiriz. Enhancing product recommender systems on sparse binary data. *Data Min. Knowl. Discov.*, 9(2):147–170, 2004.
5. Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.
6. Inderjit S. Dhillon and Yuqiang Guan. Information theoretic clustering of sparse co-occurrence data. In *Proceedings of the third International Conference on Data mining*. IEEE Press, 2003.
7. Sanjay Goil, Harsha Nagesh, and Alok Choudhary. Mafia: Efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010, Northwestern University, 2145 Sheridan Road, Evanston IL 60208, June 1999.
8. Abby Goodrum, Katherine W. McCain, Steve Lawrence, and C. Lee Giles. Scholarly publishing in the internet age: a citation analysis of computer science literature. *Information Processing and Management*, 37(5):661–675, 2001.
9. David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *J. Mach. Learn. Res.*, 1:49–75, 2001.
10. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

11. Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.
12. Zan Huang, Hsinchun Chen, and Daniel Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):116–142, 2004.
13. Bing Liu, Yiyuan Xia, and Philip S. Yu. Clustering through decision tree construction. In *Proceedings of the ninth International Conference on Information and Knowledge Management*, pages 20–29. ACM Press, 2000.
14. Sean M. McNee, Istvan Albert, Dan Cosley, Prateep Gopalkrishnan, Shyong K. Lam, Al Mamunur Rashid, Joseph A. Konstan, and John Riedl. On the recommending of citations for research papers. In *Proceedings of the 2002 ACM Conference on Computer supported cooperative work*, pages 116–125. ACM Press, 2002.
15. Stuart E. Middleton, Nigel R. Shadbolt, and David C. De Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88, 2004.
16. Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. Automatic personalization based on web usage mining. *Commun. ACM*, 43(8):142–151, 2000.
17. Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explorations.*, 6(1):90–105, 2004.
18. Anne Patrikainen and Heikki Manilla. Subspace clustering of high-dimensional binary data - a probabilistic approach. In *Workshop on Clustering High Dimensional Data and its Applications, SIAM International Conference on Data Mining.*, 2004.
19. Markus Peters and Mohammed J. Zaki. Clicks: Clustering categorical data using k-partite maximal cliques. In *IEEE International Conference on Data Engineering*. IEEE, 2005.
20. Paul Resnick and Hal R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, 1997.
21. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the tenth International Conference on World Wide Web*, pages 285–295. ACM Press, 2001.
22. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM Conference on Electronic commerce*, pages 158–167. ACM Press, 2000.

Concept Updating with Support Vector Machines

Yangguang Liu¹ and Qinming He^{1,2}

¹ College of Computer Science, Zhejiang University, Hangzhou 310027, China

² Ningbo Institute of Technology, Zhejiang University, Ningbo, 315100, China

Abstract. In many practical situations in inductive learning algorithms, it is often expected to further improve the generalization capability after the learning process has been completed if new data are available. One of the common approaches is to add training data to the learning algorithm and retrain it, but retraining for each new data point or data set can be very expensive. In view of the learning methods of human beings, it seems natural to build posterior learning results upon prior results. In this paper, we apply Support Vector Machine(SVM) to the concept updating procedure. If initial concept would be built up by inductive algorithm, then concept updated is the normal solution corresponding to the initial concept learned. It was shown that concept learned would not change if the new available data located in error-insensitive zone. Especially, concept initially learned and updated by SVR induces an incremental SVR approximately learning method for large scale data. We tested our method on toys data sets and 7 regression bench mark data set. It shown that generalization capacity after updating with SVR was improved according to FVU or MSE on the independent test set.

1 Introduction

Learning is obtaining an underlying rule by using training data sampled from the environment. Support vector machines(SVMs)[1]are trained by solving a quadratic optimization problem which needs on the order of ℓ^2 memory and time resources to solve, where ℓ is the number of training examples. In many practical situations in SVM learning, on the one hand, it is often expected to further improve the generalization capability after the learning process has been completed. One of the common approaches is to add training data to the SVM and retrain SVM, but retraining for each new data point or data set can be very expensive. In view of the learning methods of human beings, it seems natural to build posterior learning results upon prior results. Many approximate or accurate online training algorithms have been proposed for SVMs[2,3,4,5,6,7,8] to avoid retraining SVMs.

On the other hand, many applications that involve massive data sets are emerging. In order to apply SVM to large scale data problem, many researchers proposed faster implementation of SVM for classification and regression[9,10,11,12].

In this paper, we consider both sides mentioned above. we propose an incremental batch updating method which uses an incremental updating model similar to standard SVR to update the trained SVR parameters. In the next section, we formulate the standard Support Vector Machine for Regression(SVR). Then we give the updating model and explain in more detail each of its main steps and on some implementation issues. In the experiment section, we compare this new algorithm on toy and real datasets to standard batch SVR learning.

2 Support Vector Regression

Consider a training sample $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_\ell, y_\ell), \mathbf{x}_i \in R^d\}$ and $y_i \in R$, where $y_i = g(\mathbf{x}) + \zeta_i$, for some unknown function $g(\cdot)$ and noise variable ζ_i . The objective is to reconstruct a good approximation to $g(\cdot)$ from the finite data set S . The Derivation of the SVR equations can be found in [1] and will not be repeated here, but for completeness we recall some main results. Let ϕ be a nonlinear mapping from input space to some high-dimensional feature space. For the linear regressor(in feature space) defined by $f(\cdot) = \langle \mathbf{w}, \phi(\cdot) \rangle + b$, we wish to minimize

$$L(\xi, \xi^*, \mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{\ell} (\xi_i + \xi_i^*) \tag{1}$$

subject to (for $\forall i \in \{1, \dots, \ell\}$)

$$y_i - f(\mathbf{x}_i) \leq \epsilon + \xi_i^*, f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i, \xi_i, \xi_i^* \geq \epsilon, \tag{2}$$

where ϵ defines the width of the error-insensitive zone of the cost function and ξ_i and ξ_i^* are slack variables measuring the deviation of $y_i - f(\mathbf{x}_i)$ from the boundaries of the error-insensitive zone. This primal problem can be transformed to its dual quadratic optimization problem. Using the kernel trick, the solution to the dual optimization problem may be expressed solely in terms of the kernel function over the training set:

$$f(\cdot) = \sum_{i=1}^{\ell} (\alpha_i^* - \alpha_i) k(\mathbf{x}_i, \cdot) + b \tag{3}$$

where $k(\cdot, \cdot) = \langle \phi(\cdot), \phi(\cdot) \rangle$ is a kernel function, $\alpha_i^{(*)}$ is the Lagrange multiplier in the dual optimization problem.

3 Concept Updating Method

In this section, we (1) introduce an updating method for support vector regression, (2) discuss the relationship to standard SVR and show a property of proposed updating method.

3.1 Formulation of Concept Updating Method

We denoted trained SVR with parameters (\mathbf{w}_0, b_0) , and at the updating step with parameters (\mathbf{w}_k, b_k) after k times updating. We propose solving the following quadric programming problem as the updating step

$$\min_{\mathbf{w}_k, b_k} P_k = \frac{1}{2} \|\mathbf{w}_k - \mathbf{w}_{k-1}\|^2 + C \sum_{i=1}^{\ell_k} (\xi_{ki} + \xi_{ki}^*) \tag{4}$$

subject to

$$y_{ki} - (\langle \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle + b_k) \leq \varepsilon + \xi_{ki} \tag{5}$$

$$(\langle \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle + b_k) - y_{ki} \leq \varepsilon + \xi_{ki}^* \tag{6}$$

$$\xi_{ki}, \xi_{ki}^* \geq 0, i = 1, \dots, \ell_k \tag{7}$$

The same as the standard support vector machine for regression, the key idea is to construct a Lagrange function from both the objective function(it will be called the primal objective function in the rest of this article) and the corresponding constraints,by introducing a dual set of variables. It can be shown that this function has a saddle point with respect to the primal and dual variables at the optimal solution. We construct the Lagrange function as follows:

$$\begin{aligned} L_k &= \frac{1}{2} \|\mathbf{w}_k - \mathbf{w}_{k-1}\|^2 + C \sum_{i=1}^{\ell_k} (\xi_{ki} + \xi_{ki}^*) \tag{8} \\ &\quad - \sum_{i=1}^{\ell_k} (\eta_{ki} \xi_{ki} + \eta_{ki}^* \xi_{ki}^*) \\ &\quad - \sum_{i=1}^{\ell_k} \alpha_{ki} (\varepsilon + \xi_{ki} + y_{ki} - \langle \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle - b_k) \\ &\quad - \sum_{i=1}^{\ell_k} \alpha_{ki}^* (\varepsilon + \xi_{ki}^* - y_{ki} + \langle \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle + b_k) \end{aligned}$$

It is understood that the dual variables in (8) have to satisfy positivity constraints, i.e. $\alpha_{ki}, \alpha_{ki}^*, \eta_{ki}, \eta_{ki}^* \geq 0$. It follows from the saddle point condition that the partial derivatives of L_k with respect to the primal variables $\mathbf{w}_k, b_k, \xi_{ki}, \xi_{ki}^*$ have to vanish for optimality.

$$\partial_{b_k} L_k = \sum_{i=1}^{\ell_k} (\alpha_{ki}^* - \alpha_{ki}) = 0 \tag{9}$$

$$\begin{aligned} \partial_{\mathbf{w}_k} L_k &= \mathbf{w}_k - \mathbf{w}_{k-1} - \sum_{i=1}^{\ell_k} (\alpha_{ki}^* - \alpha_{ki}) \phi(\mathbf{x}_{ki}) \\ &= 0 \end{aligned} \tag{10}$$

$$\partial_{\xi_{ki}^{(*)}} L_k = C - \alpha_{ki}^{(*)} - \eta_{ki}^{(*)} = 0 \tag{11}$$

Substituting (9),(10), and (11) into (8) yields the dual optimization problem.

$$\begin{aligned} \min_{\alpha_k, \alpha_k^*} D &= \frac{1}{2} \sum_{i,j=1}^{\ell_k} K_{ij}(\alpha_{ki} - \alpha_{ki}^*)(\alpha_{kj} - \alpha_{kj}^*) \\ &+ \varepsilon \sum_{i=1}^{\ell_k} (\alpha_{ki} + \alpha_{ki}^*) \\ &- \sum_{i=1}^{\ell_k} (y_{ki} - f_{k-1}(\mathbf{x}_{ki}))(\alpha_{ki} - \alpha_{ki}^*) \end{aligned} \tag{12}$$

subject to

$$0 \leq \alpha_{ki}, \alpha_{ki}^* \leq C, i = 1, \dots, \ell_k, \tag{13}$$

$$\sum_{i=1}^{\ell_k} (\alpha_{ki} - \alpha_{ki}^*) = 0 \tag{14}$$

In deriving (12), (13), and (14) we already eliminated the dual variables $\eta_{ki}^{(*)}$ through condition (11), as these variables did not appear in the dual objective function anymore but only were present in the dual feasibility conditions. Eq. (10) can be rewritten as follows

$$\mathbf{w}_k = \mathbf{w}_{k-1} + \sum_{i=1}^{\ell_k} (\alpha_{ki}^* - \alpha_{ki}) \phi(\mathbf{x}_{ki}) \tag{15}$$

and therefore

$$f_k(\mathbf{x}) = \sum_{i=1}^{\ell_k} (\alpha_{ki}^* - \alpha_{ki}) \langle \phi(\mathbf{x}_{ki}), \phi(\mathbf{x}) \rangle + f_{k-1}(\mathbf{x}) + b_k - b_{k-1} \tag{16}$$

From above, it is known that \mathbf{w}_k can be completely described as sum of \mathbf{w}_{k-1} and new available training patterns \mathbf{x}_{ki} . The bias term b_k follows from KKT conditions which is not further discussed here.

3.2 Updating Procedure

In fact, the updating procedure can be described as following

1. Learning initial concept: training SVR with initial data, and then attain the initial regression function determined by \mathbf{w}_0, b_0
2. Organizing new data: There exist many strategies to organize the available data , for example, we can use all available data to update learned concept or only use part data which satisfy some special condition. In our experiments, we use all available data to update learned concept.
3. updating last learned concept: solve updating problem with organized data, then attain the updated regression function determined by (\mathbf{w}_k, b_k) . Return last step if new data available.

3.3 Relation to Standard SVR

Let $\Delta \mathbf{w}_k = \mathbf{w}_k - \mathbf{w}_{k-1}$, $\Delta y_{ki} = y_{ki} - f_{k-1}(\mathbf{x}_{ki})$, $\Delta b_k = b_k - b_{k-1}$, then $\mathbf{w}_k = \mathbf{w}_{k-1} + \Delta \mathbf{w}$. So we can rewrite the updating formula as follows.

$$\min_{\mathbf{w}_k, b_k} P_k = \frac{1}{2} \|\Delta \mathbf{w}_k\|^2 + C \sum_{i=1}^{\ell_k} (\xi_{ki} + \xi_{ki}^*) \tag{17}$$

subject to

$$\Delta y_{ki} - (\langle \Delta \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle + \Delta b_k) \leq \varepsilon + \xi_{ki} \tag{18}$$

$$(\langle \Delta \mathbf{w}_k, \phi(\mathbf{x}_{ki}) \rangle + \Delta b_k) - \Delta y_{ki} \leq \varepsilon + \xi_{ki}^* \tag{19}$$

$$\xi_{ki}, \xi_{ki}^* \geq 0, i = 1, \dots, \ell_k \tag{20}$$

From above rewriting formula, we can see that updating method has the same formula as SVR, but now the training data are $\{(\mathbf{x}_{k1}, \Delta y_{k1}), \dots, (\mathbf{x}_{ki}, \Delta y_{ki})\}$.

The following theorem shows that updating model do not change the learned concept or its representation of the support vectors if new available data locate in the ε -tube.

Theorem 1. *At the k -th learning step, if the positive definite kernel function and its parameters keep the same as the $(k-1)$ -th step, kernel function is strictly positive definition function, and new data set D_k satisfies condition*

$$|y_{ki} - f_{k-1}(\mathbf{x}_{ki})| \leq \varepsilon, \quad i = 1, \dots, \ell_k \tag{21}$$

then $\alpha_{ki}^{(*)} = 0, i = 1, \dots, \ell_k$.

Proof. Obviously $\alpha_{ki}^{(*)} = 0, i = 1, \dots, \ell_k$ is a feasible solution to the SVPL dual problem. The following proof it is also unique optimal solution. Considering (12), We decomposed it into two items $D = D_1 + D_2$, where

$$D_1 = \frac{1}{2} \sum_{i,j=1}^{\ell_k} K_{ij} (\alpha_{ki} - \alpha_{ki}^*) (\alpha_{kj} - \alpha_{kj}^*)$$

$$D_2 = \varepsilon \sum_{i=1}^{\ell_k} (\alpha_{ki} + \alpha_{ki}^*) - \sum_{i=1}^{\ell_k} (y_{ki} - f_{k-1}(\mathbf{x}_{ki})) (\alpha_{ki} - \alpha_{ki}^*)$$

According to the definition of positive definition function, then $D_1 \geq 0$, and if kernel function K is a strictly positive definition function, inequality holds strictly. And inequality $D_2 \geq 0$ holds under the condition(21), since we can rewrite the D_2 as follows

$$D_2 = \sum_{i=1}^{\ell_k} (\varepsilon - y_{ki} + f_{k-1}(\mathbf{x}_{ki})) \alpha_{ki} + \sum_{i=1}^{\ell_k} (\varepsilon + y_{ki} - f_{k-1}(\mathbf{x}_{ki})) \alpha_{ki}^*$$

Considering the theorem condition(21) and $\alpha_{ki}^{(*)} \geq 0, i = 1, \dots, \ell_k$. It follows $D_2 \geq 0$.

According to property of strictly convex quadratic programming and $D \geq 0$, then $\alpha_{ki}^{(*)} = 0, i = 1, \dots, \ell_k$ also is unique and optimal solution. ■

4 Experiments

In this section, we report the evaluation results. The updating problem is solved based on the LibSVM¹. All our experiments are done in a Celoren 1.7Ghz machine with 256MB memory running on Windows2000.

4.1 Evaluations on Synthetic Toy Data Sets

We demonstrate the advantages of our approach in comparison with standard SVR in following synthetic toy data sets first.

We investigate the updating method on six nonlinear function $g^{(j)}, j = 1, \dots, 6$. The function $g^{(1)} : [0, 1] \rightarrow R$, and remaining five nonlinear function $g^{(j)} : [0, 1] \rightarrow R, j = 2, \dots, 6$, which have been used in paper [13]. Their equations are given below(see Fig. 1):

1. Sinc Function

$$g^{(1)}(x) = sinc(x)$$

2. Simple Interaction Function

$$g^{(2)}(x_1, x_2) = 10.391((x_1 - 0.4) \cdot (x_2 - 0.6) + 0.36)$$

3. Radial Function

$$g^{(3)}(x_1, x_2) = 24.234(r^2(0.75 - r^2)), r^2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2$$

4. Harmonic Function

$$g^{(4)}(x_1, x_2) = 42.659((2 + x_1)/20 + Re(z^5))$$

where $z = x_1 + ix_2 - 0.5(1 + i)$, or, equivalently, with $\bar{x}_1 = x_1 - 0.5, \bar{x}_2 = x_2 - 0.5$,

$$g^{(4)}(x_1, x_2) = 42.659(0.1 + \bar{x}_1(0.05 + \bar{x}_1^4 - 10\bar{x}_1^2\bar{x}_2^2 + 5\bar{x}_2^4))$$

5. Additive Function

$$g^{(5)}(x_1, x_2) = 1.3356(1.5(1 - x_1) + e^{2x_1-1} \sin(3\pi(x_1 - 0.6)^2) + e^{3(x_2-0.5)} \sin(4\pi(x_2 - 0.9)^2))$$

6. Complicated Interaction Function

$$g^{(6)}(x_1, x_2) = 1.9(1.35 + e^{x_1} \sin(13(x_1 - 0.6)^2)e^{-x_2} \sin(7x_2))$$

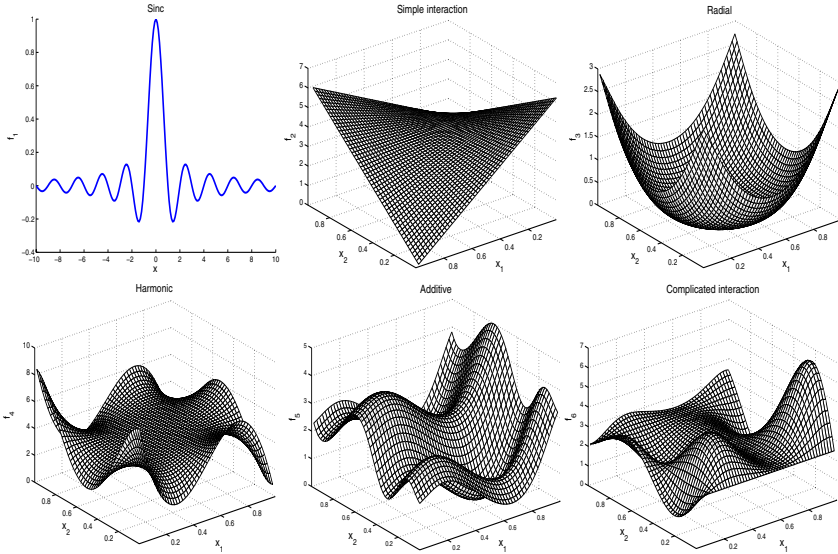


Fig. 1. Noiseless case: Six nonlinear functions for the updating method

Training Data: We generate ten different data sets which have 250 pairs (x_{i1}, x_{i2}) , among which 150 for initial SVR training, 100 pairs for updating, with each component an independent uniform random variable on $[0, 1]$. And then run the algorithms twice for each function, once in noiseless situation (see Fig. 1): $y_i^{(j)} = g^{(j)}, i = 1, \dots, 250, j = 1, \dots, 6$, and once in the slightly noisy situation (see Fig. 2) $y_i^{(j)} = g^{(j)} + 0.25\epsilon_i, i = 1, \dots, 250, j = 1, \dots, 6$, where the ϵ_i 's are *i.i.d* $N(0, 1)$.

Parameter Settings: In our experiments, both standard SVR and updating method use Gaussian kernel with width parameter $\sigma = 1$, regularized parameter $C = 1$ and all other parameters are the default parameters in LibSVM tools.

Experimental Results: To test the accuracy of the updating method, a test set of size $N = 10,000$ was generated over a regular grid on $[0, 1]$, or $[0, 1]^2$, with $x_{ir} = (2i - 1)/200, i = 1, \dots, 100, r = 1, 2$. The error measured is the fraction of variance of unexplained (FVU), which is given by

$$FVU = \frac{E(\hat{g}(\mathbf{x})_i - g(\mathbf{x}_i))^2}{E(g(\mathbf{x})_i - \bar{g}(\mathbf{x}_i))^2}.$$

where $g(\mathbf{x}_i)$ is the true value of the function and $\hat{g}(\mathbf{x}_i)$ is the estimated value. We evaluate the FVU, which is the mean squared error for the estimate $\hat{g}(\mathbf{x})$ scaled by the variance of the true function $g(\mathbf{x})$, for an estimator by replacing the expectations with averages over the grid of test set values.

¹ Available at <http://www.csie.ntu.edu.tw/~cjlin/libSVM>

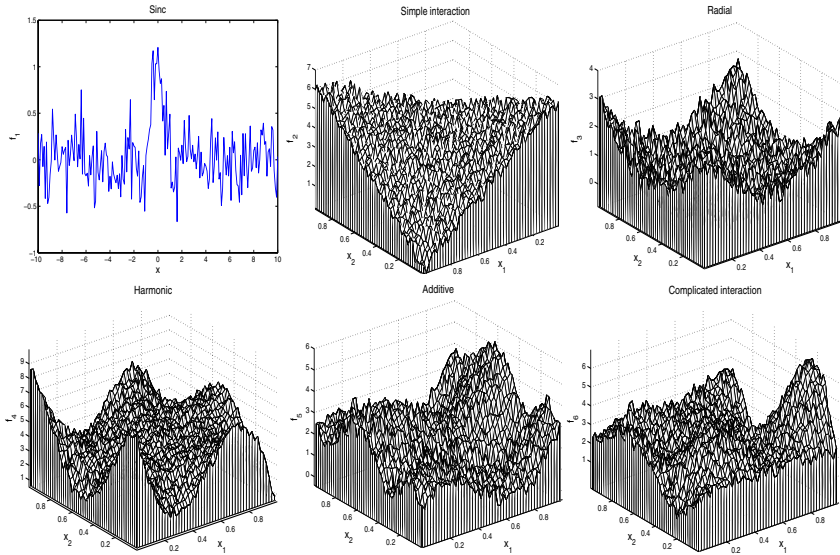


Fig. 2. Noisy case: Six nonlinear functions for the updating method

Results for the noiseless and noisy situations are presented in Table 1. From the results, we observe that updating procedure improve the performance on the all cases. From the results, we observe that updating procedure improve the performance on the all cases. In comparison with standard batch SVR learning, updating method wins five noisy cases, but half noiseless cases.

Table 1. FVU of standard SVR and updating method

Function	noiseless data			noisy data		
	SVR	Initial SVR	Updating	SVR	Initial SVR	Updating
Sinc	0.00117	0.00452	0.00220	0.431	0.583	0.517
Simp Int	0.00035	0.00057	0.00028	0.00896	0.01466	0.00767
Radial	0.00973	0.01991	0.00996	0.03162	0.05612	0.03072
Harmonic	0.07478	0.08942	0.06677	0.25389	0.36352	0.20851
Additive	0.00101	0.00186	0.00112	0.01859	0.02099	0.01841
Comp Int	0.04891	0.05381	0.04537	0.14633	0.26950	0.04272

4.2 Evaluations on Other Data Sets

We perform evaluations on seven standard data sets: Abalone, Census-house (house-price-16H, house-price-16L, house-price-8H, house-price-8L) and Computer system activity data (cpu, cpu-small). All the data sets were real data from the UCI machine learning repository. We compared updating method with

Table 2. Experimental results

Data set	Initial MSE	Updating MSE	SVR MSE	#iter SVR	#iter upd.
Adalone	4.4768	4.37	4.42	69058	52793
House-16H	1.39E+10	1.38E+10	1.39E+10	35219	9157
House-16L	4.76E+09	2.10E+09	4.57E+09	9492	4752
House-8H	2.61E+09	2.49E+09	2.53E+09	9505	4546
House-8L	4.92E+09	2.12E+09	4.58E+09	11062	4598
CPU	15.35	13.243	11.78	794270	406711
CPU-Small	9.96	10.10	9.86	837169	401446

standard SVR with Gaussian kernel. The parameter C for both updating and standard SVR was tuned via cross validations, so was the width parameter in the Gaussian kernel for SVR. We use one of three training data for updating and keep the the parameters unchange during updating. The final performance results were obtained via the 10-fold cross validation. Table2 summarizes the evaluation results.

From the results, we observe that updating procedure improve the performance and decrease the iterative number of algorithm dramatically on the all cases. In comparison with standard SVR, updating method wins five cases in the Gaussian kernel based on the minimized square errors(MSE) on independent test set. And The evaluations on these standard bench-mark data sets demonstrate that it is worth considering available data to updating trained SVR.

5 Conclusion

We propose a novel updating method for SVR. It is shown that updating procedure is to solve an convex quadratic programming the same as the standard support vector machine algorithm, and if new available data locate in error insensitive zone, then learned concept and his representation of support vector will not change. Comparison to the SVR trained with all data in one batch, experimental results show very promising results and open the door to real applications.

Acknowledgments

This research was supported by Ningbo Doctor Science Fund grant 2005A610002.

References

1. Vapnik, V.: Statistical Learning Theory, Wiley, New York(1998)
2. Syed, N.A., Liu, H., Sung, K.: Handling concept drifts in incremental learning with support vector machines, In Proceedings of 1st Intl. Conf. on Knowledge Discovery and Data Mining, AAAI(1999) 317-321

3. Ralaivola, L., d'Alch-Buc F.: Incremental Support Vector Machine Learning: A Local Approach. Georg Dorffner, Horst Bischof, Kurt Hornik (Eds.): Lecture Notes in Computer Science 2130, Vienna, Austria: Springer (2001) 322–330
4. Engel Y., Mannor S. and Meir R.: Sparse online greedy support vector regression. In Elomaa, T., and Hannu Toivonen, H. M., eds., ECML, Vol. 2430 of Lecture Notes in Computer Science, 84–96, Springer(2002)
5. Herbster M.: Learning additive models online with fast evaluating kernel. In Proceedings of 14th Annual Conference on Computational Learning Theory (COLT), pp. 444–460. Springer(2001)
6. Kivinen J., Smola A. J. and Williamson R. C.: Online learning with kernel. In Dietterich, T. G.; Becker, S.; and Ghahramani, Z., eds., Advances in Neural Information Processing Systems, (2001) 785–792
7. Li Y. and Long P.: The relaxed online maximum margin algorithm. In Solla, A.; Leen, T. K.; and Miller, K.- R., eds., Advances in Neural Information Processing Systems, Vol. 12, pp. 498–504. Cambridge, MA: MIT Press(2001)
8. Ma J., Theiler J. and Perkins S.: Accurate online support vector regression. Neural Computation, Vol.15 No. 11(2003) 2683–2703
9. Collobert R. and Bengio S.: SVM-Torch: Support vector machines for large-scale regression problems, Journal of Machine Learning Research, Vol. 1 143–160(2001)
10. Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. In Proceedings, 1997 IEEE Workshop on Neural Networks for Signal Processing, (1997) 276–285.
11. Flake G. and Lawrence S.: Efficient svm regression training with smo. Machine Learning, Vol. 46(2002) 271–290
12. Joachims T.: Making large-scale support vector machine learning practical. In Scholkopf, B.; Burges, C.; and Smola, A., eds., Advances in Kernel Methods. MIT Press(1999)
13. Roosen C.B. and Hastie T.J.: Logistic response projection pursuit, Tech. Rep. BL011214-930806-09TM, AT&T Bell Laboratories (1993)

On the Performance of Feature Weighting K -Means for Text Subspace Clustering

Liping Jing^{1,2}, Michael K. Ng³, Jun Xu², and Joshua Zhexue Huang²

¹ Department of Mathematics, The University of Hong Kong, Hong Kong, China

² E-Business Technology Institute, The University of Hong Kong, Hong Kong, China
{lpjing, fxu, jhuang}@eti.hku.hk

³ Department of Mathematics, Hong Kong Baptist University, Hong Kong, China

Abstract. Text clustering is an effective way of not only organizing textual information, but discovering interesting patterns. Most existing methods, however, suffer from two main drawbacks; they cannot provide an understandable representation for text clusters, and cannot scale to very large text collections. Highly scalable text clustering algorithms are becoming increasingly relevant. In this paper, we present a performance study of a new subspace clustering algorithm for large sparse text data. This algorithm automatically calculates the feature weights in the k -means clustering process. The feature weights are used to discover clusters from subspaces of the text vector space and identify terms that represent the semantics of the clusters. A series of experiments have been conducted to test the performance of the algorithm, including resource consumption and clustering quality. The experimental results on real-world text data have shown that our algorithm quickly converges to a local optimal solution and is scalable to the number of documents, terms and the number of clusters.

Keywords: Subspace Clustering, Text Clustering, Feature Weighting, Scalability, Convergency.

1 Introduction

Clustering text documents into different category groups is an important step in indexing, retrieval, management and mining of abundant text data on the Web or in corporate information systems. Many algorithms to address clustering problems have been developed according to the specific domain requirements [1]. Usually, the distance or dissimilarity measures of these algorithms involve all features of the data set [2,3]. This is applicable only if all or most features are important to every cluster. However, in text documents, clusters are often discovered in different feature subspaces, i.e., different clusters have different subsets of important features or key words.

To effectively cluster large and sparse text data requires the clustering algorithms to be efficient, scalable and able to discover clusters from subspaces of the text vector space model (*VSM*). Scalable subspace clustering methods are

made good candidates for text clustering [4], while other clustering algorithms often fail to produce satisfactory clustering results. Many existing subspace clustering algorithms [5] are designed for structured data, so they can not effectively address the following two special issues: high dimensionality and sparsity, which are problematic in text mining.

In order to address the problems mentioned above, a new subspace clustering algorithm with feature weighting capability was studied by Jing et al. [6]. We denoted it as *FW-KMeans*. This algorithm can automatically compute feature weights based on the importance of the features in clustering. The more important the term for one cluster, the higher the weight. A weight calculation formula that minimizes the cost function of clustering is provided. This algorithm is aimed to solve the challenging problems of big volume and high dimensionality in text clustering. In the previous work [6], we have demonstrated that *FW-KMeans* outperformed the *Bisection KMeans* [7] and *Standard KMeans* algorithms. The main contribution of this work is to present some theoretical explanations and empirical tests of the *FW-KMeans* algorithm. The empirical tests were conducted with the real-world text data set 20-Newsgroups. We used the unified preprocessing steps for all experiments. The main results obtained can be summarized as follows:

- **Convergency:** *FW-KMeans* quickly converges to a local optimal solution, no matter how many documents, terms and clusters are;
- **Scalability:** *FW-KMeans* is scalable [8] to the number of documents, terms and clusters. The run-time increases linearly as the number of documents, terms and clusters increases.

The rest of the paper is organized as follows. Section 2 describes subspace clustering with the feature weighting *k*-means algorithm. Section 3 analyzes the algorithm in terms of resource consumption. Following the discussion of our method, we present, in Section 4, the empirical results of our experiments in both scalability and convergency. Finally, we draw some conclusions and point out our future work in Section 5.

2 *K*-Means with Feature Weighting

In our research, documents are represented using a *bag-of-words* representation [9]. In this representation (also named as *VSM*), a set of documents are represented as a set of vectors $\mathbf{X} = \{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_n\}$. Each vector \mathcal{X}_j is characterized by a set of m terms or words, (t_1, t_2, \dots, t_m) . Here, the terms can be considered as the features of the vector space and m as the number of dimensions representing the total number of terms in the vocabulary. Assume that several categories exist in \mathbf{X} , each category of documents is characterized by a subset of terms in the vocabulary that corresponds to a subset of features in the vector space. In this sense, we say that a cluster of documents is situated in a subspace of the vector space.

2.1 Subspace Clustering

A simple example is given in Table 1 to show that it is necessary and effective to use subspace clustering on text data. Here, x_j represents the j th document

Table 1. An example of a data set in *VSM*

		t_0	t_1	t_2	t_3	t_4
C_0	x_0	1	2	3	0	2
	x_1	2	3	1	0	2
	x_2	3	1	2	0	2
C_1	x_3	0	0	1	3	2
	x_4	0	0	2	1	3
	x_5	0	0	3	2	1

vector; t_i represents the i th term; each cell in the table is the frequency that term t_i occurs in x_j .

We can see that there are many zero entries in the table. A zero entry means that the term does not appear in the related document. Table 1 shows that the number of terms covered by each document is much smaller than the total number of terms in the vocabulary. For instance, t_3 does not occur in cluster C_0 ; and so do t_0 and t_1 in cluster C_1 , which results in the special characteristic of text data: sparsity. Meanwhile, this situation requires text clustering to be focused on subspace instead of the entire space. For example, cluster C_0 can be represented by t_0 , t_1 , t_2 and t_4 without t_3 ; cluster C_1 represented only with t_2 , t_3 and t_4 . Additionally, in text clustering, a good term or word will appear in the majority of the documents of a cluster with similar frequency, such as that t_4 appears two times in every document of cluster C_0 . Therefore, we cannot simply think that all terms in each cluster play an equally important role in the clustering process. For example, t_4 should be more important than t_0 , t_1 and t_2 in cluster C_0 . In order to deal with these special issues, we introduced feature weighting k -means for text subspace clustering [6].

Feature weighting k -means finds the weight for each feature from each cluster. Let $A = (A_1, A_2, \dots, A_k)$ be the set of weight vectors for all clusters; $A_l = (\lambda_{l,1}, \lambda_{l,2}, \dots, \lambda_{l,m})$ be the weights for m features from the l th cluster. During the k -means clustering process, our method automatically calculates the feature weights which produces a $k \times m$ weights matrix. That is to say, in each cluster m weights are assigned to m features. According to the above analysis, a larger weight will be produced for the term that appears in the majority of the documents of one cluster with similar frequency, which means that the weight of a feature is inversely proportional to the dispersion of values of that feature. The larger the dispersion, the smaller the weight. This indicates that the values of a good feature in a cluster are very close to the value of the cluster center in that feature. For example in Table 1, a larger weight will be assigned to t_4 for cluster C_0 . However, feature t_3 does not appear in cluster C_0 , which implicitly means

that feature t_3 has a smaller dispersion and will also get a larger weight. To identify the cluster, term t_4 is apparently more important than term t_3 . Fortunately, the two different terms can be easily separated in post-processing. When extracting important features to represent different clusters, t_3 type features will be removed but t_4 type features will be retained.

In order to balance the features that never appear in one cluster, we introduce a parameter σ . The problem of clustering \mathbf{X} into k clusters can be stated as an algorithm which attempts to minimize the following cost function:

$$F(W, Z, \Lambda) = \sum_{l=1}^k \sum_{j=1}^n \sum_{i=1}^m w_{l,j} \lambda_{l,i}^\beta [d(z_{l,i}, x_{j,i}) + \sigma] \quad (1)$$

subject to

$$\begin{cases} \sum_{l=1}^k w_{l,j} = 1, & 1 \leq j \leq n \\ w_{l,j} \in \{0, 1\}, & 1 \leq j \leq n, \quad 1 \leq l \leq k \\ \sum_{i=1}^m \lambda_{l,i} = 1, & 0 \leq \lambda_{l,i} \leq 1, \quad 1 \leq l \leq k \end{cases} \quad (2)$$

where $k(\leq n)$ is a known number of clusters, β is an exponent greater than 1 [10]; $W = [w_{l,j}]$ is a $k \times n$ integer matrix; $Z = [Z_1, Z_2, \dots, Z_k] \in \mathbf{R}^{k \times m}$ represents k cluster centers; $d(z_{l,i}, x_{j,i}) (\geq 0)$ is a distance or dissimilarity measure between object j and the centroid of cluster l on the i th feature. Usually, we use Euclidean distance:

$$d(x_{j,i}, z_{l,i}) = (x_{j,i} - z_{l,i})^2 \quad (3)$$

The value of parameter σ will affect the feature weighting process. If σ is much larger than $d(\tilde{z}_{l,i}, x_{j,i})$, the weights will be dominated by σ and $\lambda_{l,i}$ will approach to $\frac{1}{m}$. This will make the clustering process back to the standard k -means. If σ is too small, then the gap of the weights between the zero dispersion features and other important features will be big, therefore, undermining the importance of other features. To balance we calculate σ based on the average dispersion of the entire data set for all features as follows:

$$\sigma = \frac{\sum_{j=1}^{\hat{n}} \sum_{i=1}^m d(x_{j,i}, o_i)}{\hat{n} \cdot m} \quad (4)$$

where o_i is the mean feature value of the entire data set. In practice we use a sample instead of the entire data set to calculate σ . (5% sample is used according to the sampling theory [11].) \hat{n} is the number of documents in the sample. Experimental results in [6] have shown that this selection of σ is reasonable to produce satisfactory clustering results and identify important features of clusters.

3 *FW-KMeans* Algorithm Analysis

In order to solve the minimization problem described in Section 2, we designed a subspace clustering algorithm, denoted as: *FW-KMeans*.

1. Choose initial cluster centers $Z^{(1)} \in \mathbf{R}^{k \times m}$ and set $\Lambda^{(1)}$ with all entries equal to $1/m$. Set $h = 1$.
2. Use Eq.5 to calculate $W^{(h+1)}$. If $F(W^{(h+1)}, Z^h, \Lambda^h) = F(W^h, Z^h, \Lambda^h)$ then stop; otherwise $h = h + 1$, goto step 3.
3. Use Eq.6 to calculate $Z^{(h+1)}$. If $F(W^{(h+1)}, Z^{(h+1)}, \Lambda^h) = F(W^{(h+1)}, Z^h, \Lambda^h)$ then stop; otherwise $h = h + 1$, goto step 4.
4. Use Eq.7 to calculate $\Lambda^{(h+1)}$. If $F(W^{(h+1)}, Z^{(h+1)}, \Lambda^{(h+1)}) = F(W^{(h+1)}, Z^{(h+1)}, \Lambda^h)$ then stop; otherwise $h = h + 1$, goto step 2.

FW-KMeans is a solution to the above constrained nonlinear optimization problem, which can be solved by iteratively executing Step 2, 3 and Step 4. The whole clustering process can be divided into three steps to optimize W, Z and Λ respectively. The usual method towards optimization of $F(\cdot, \cdot, \cdot)$ is to use partial optimization for W, Z and Λ . In this method we first fix Z and Λ , and find necessary conditions on W to minimize $F(\cdot, \cdot, \cdot)$. Then we fix W and Λ , and minimize $F(\cdot, \cdot, \cdot)$ with respect to Z . We then fix W and Z , and minimize $F(\cdot, \cdot, \cdot)$ with respect to Λ . The process is repeated until no more improvement in the objective function value can be made.

The matrices W, Z and Λ can be calculated with the following equations: When fixing \tilde{Z} and $\tilde{\Lambda}$, the partition matrix W is computed by

$$\begin{cases} w_{l,j} = 1 & \text{if} & \sum_{i=1}^m \lambda_{l,i}^\beta (d(z_{l,i}, x_{j,i}) + \sigma) \\ & & \leq \sum_{i=1}^m \lambda_{h,i}^\beta (d(z_{h,i}, x_{j,i}) + \sigma) \text{ for } 1 \leq h \leq k \\ w_{l,j} = 0 & \text{otherwise} \end{cases} \quad (5)$$

The proof of Eq.5 can be found in [12]. We remark that the minimum solution \tilde{W} is not unique, so $w_{l,j} = 1$ may arbitrarily be assigned to the first minimizing index l , and the remaining entries of the column are put to zero. And $w_{l,j} = 1$ means that the j th document belongs to the l th cluster because they have the smallest distance.

When fixing \tilde{W} and $\tilde{\Lambda}$, the cluster center vectors set Z in each iteration is updated by

$$z_{l,i} = \frac{\sum_{j=1}^n w_{l,j} x_{j,i}}{\sum_{j=1}^n w_{l,j}} \quad \text{for } 1 \leq l \leq k \text{ and } 1 \leq i \leq m \quad (6)$$

When fixing \tilde{W} and \tilde{Z} , the feature weighting matrix Λ is calculated by

$$\lambda_{l,i} = \frac{1}{\sum_{t=1}^m \left[\frac{\sum_{j=1}^n \tilde{w}_{l,j} [d(\tilde{z}_{l,i}, x_{j,i}) + \sigma]}{\sum_{j=1}^n \tilde{w}_{l,j} [d(\tilde{z}_{l,t}, x_{j,t}) + \sigma]} \right]^{1/(\beta-1)}} \quad (7)$$

The above algorithm converges to a local minimal solution in a finite number of iterations. This can be proved as follows. Firstly, we note that there are only a

finite number of possible partitions W . We then show that each possible partition W appears at most once by the algorithm. Assume that $W^{h_1} = W^{h_2}$ where $h_1 \neq h_2$. We note that given W^h , we can compute the minimizer Z^h which is independent of W^h according to Eq.6. For W^{h_1} and W^{h_2} , we have the minimizers Z^{h_1} and Z^{h_2} respectively. It is clear that $Z^{h_1} = Z^{h_2}$ since $W^{h_1} = W^{h_2}$. Using W^{h_1} and Z^{h_1} , and W^{h_2} and Z^{h_2} , we can compute the minimizers Λ^{h_1} and Λ^{h_2} respectively (Step 4) according to Eq.7. Again, $W^{h_1} = W^{h_2}$. Therefore, we have

$$F(W^{h_1}, Z^{h_1}, \Lambda^{h_1}) = F(W^{h_2}, Z^{h_2}, \Lambda^{h_2}).$$

However, the sequence $F(\cdot, \cdot, \cdot)$ generated by the algorithm is strictly decreasing. Hence the result follows.

Since the *FW-KMeans* algorithm is an extension to the k -means algorithm by adding a new step to calculate the feature weights in the iterative process, it does not seriously affect the scalability of the k -means type algorithms in clustering large data. An analysis of the run-time complexity for our algorithm shows this result. There are essentially three major computation steps in the algorithm and the complexity for each step is discussed in turn:

1. *Partitioning Documents*

After initialization of feature weights and cluster centers in Step 1, A cluster label will be assigned to each document of the data set. This process simply compares the summation, $\sum_{i=1}^m \lambda_{l,i}^\beta (d(z_{l,i}, x_{j,i}) + \sigma)$, for each document in all k clusters. Thus, the complexity for this step is $O(mnk)$.

2. *Updating Cluster Centers*

With the partition matrix W , the task of updating cluster centers is to find the mean of the documents which belong to the same cluster. Thus, for k clusters, the run-time complexity for this step is $O(mnk)$.

3. *Calculating Feature Weights*

The last phase of our algorithm is to assign feature weights for all clusters based on the partition matrix W . Since in our current implementation, we only go through the whole data set once to update the feature weights, therefore, the runtime complexity of this step is also $O(mnk)$.

Therefore, the total runtime complexity of our algorithm is $O(hmnk)$, where h is the total number of iterations.

As for the storage, we need $O(2mk)$ space to hold the cluster centers Z and the feature weighting matrix Λ ; also $O(n(1 + \bar{m}))$ space to store the set of n documents and their cluster labels, where \bar{m} is the average number of terms in each document and much smaller than m , because we only store the nonzero entries as shown in Table 1 in order to save the memory consumption.

4 Experiments

In this section, we use experimental results to demonstrate the clustering performance of the *FW-KMeans* algorithm. We have done experiments on real-world

text data sets to test the clustering quality in discovering clusters and identifying significant features from given data sets for each cluster [6]. Here, we conducted a scalability and convergency benchmark test of the algorithm on large text data sets.

We have used the clustering accuracy as a measure of clustering quality. Clustering accuracy is defined as

$$Accuracy = \frac{\sum_{l=1}^k C_l}{n}$$

where C_l is the number of documents occurring in both cluster l and its corresponding category and n is the number of documents in the data set.

4.1 Text DataSets

Fourteen datasets were built from the *20-Newsgroups*¹ collection with different characteristics in size, dimensionality and class distribution.

Firstly, the number of terms (m) increased 300 in each step from 500 to 2000, while holding the number of documents (n) to 15905 and the number of clusters (k) to 20. In this way, we generated one data collection of 6 data subsets. When choosing the terms, we adopted the feature selection method [13] to calculate the feature values, and sort all terms according to these values. Then we selected the first m terms with the relatively higher scores.

Secondly, we fixed the number of terms to 1100 and the number of clusters to 20, and then increased the number of documents from 2000 to 15905 with the rate of two times per step. Another data collection containing 4 data subsets was generated. Similarly, we only choose the first 1100 terms with the higher scores to represent the document vectors. Meanwhile, the two data collections overlap on the subset (15905, 1100, 20) (15905 documents, 1100 terms and 20 clusters), and cover all topics in *20-Newsgroups*.

Finally, we created our last data collection with 5 data sets. When creating this data collection, we made the number of clusters varying in {3, 5, 7, 10, 12}, while fixing the number of documents to 1500 and always extracting the first 500 terms with higher scores. For all data collections, we used the documents from the following 12 topics: *alt.atheism*, *comp.graphics*, *talk.politics.guns*, *rec.autos*, *soc.religion.christian*, *misc.forsale*, *sci.crypt*, *comp.sys.ibm.pc.hardware*, *rec.sport.basketball*, *sci.space*, *comp.os.windows*, and *talk.politics.mideast*. By varying the number of clusters (k), we selected documents from the first k topics.

The raw data were preprocessed using the *BOW* toolkit [14]. The pre-processing steps include removing the headers, the stop words, and the words that occur in less than three documents or greater than the average number of documents in each class, as well as stemming the left words with the Porter stemming function. The standard term scoring method

$$tfidf(d, t) = tf(d, t) \times \log(n/df(t))$$

¹ <http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html>

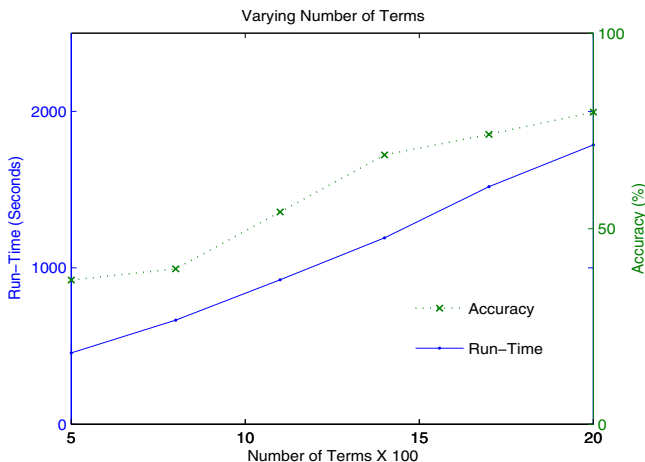


Fig. 1. Run-time and accuracy for different numbers of terms

was used to represent the document vector, where $tf(d, t)$ is the term t frequency in document d , and $df(d, t)$ is the document frequency of term t that counts the number of documents term t appears.

4.2 Experimental Results

We used the datasets described above to conduct a benchmark test on the scalability of *FW-KMeans*. The experiments were done on a machine with a 3.2G CPU and 2G RAM. The parameter β was assigned to 1.5 in all experiments. The run-time and clustering accuracy of the *FW-KMeans* algorithm were experimented with respect to the size, dimensionality and class distribution. All experiments were repeated five times and the running time represents the average over five runs.

Fig.1 shows the clustering accuracy and run-time (in seconds) when the number of terms varied from 500 to 2000, while the number of documents was fixed to 15905 and the number of clusters to 20. We can notice that the run-time linearly increased with the number of terms, which reflexes the linear computational complexity $O(hmnk)$ analyzed in Section 3. The accuracy (i.e., clustering quality) increased a little when the number of terms increased, because more terms held more information about the raw text, therefore, the clustering quality was improved.

Fig.2 shows the run-time and clustering accuracy when the number of documents varied from 2000 to 15905, while the number of terms was fixed to 1100 and the number of clusters to 20. The iteration h changed in $\{9,13,13,10\}$ when the algorithm converged to a local optimal solution for each data set. The run-time increased linearly with the number of documents, which also verified the linear computational complexity $O(hmnk)$ analyzed in Section 3. The clustering

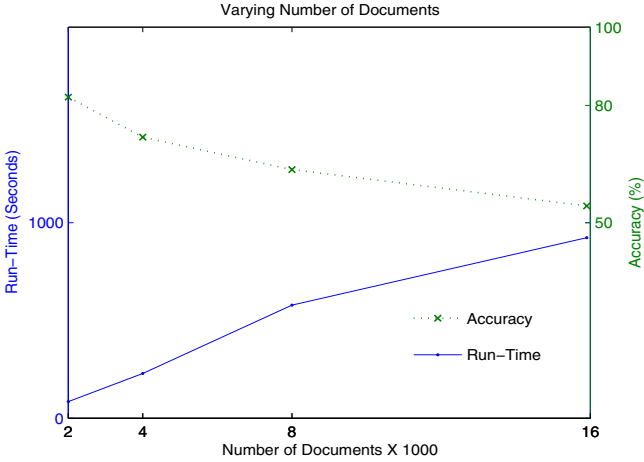


Fig. 2. Run-time and accuracy for different numbers of documents

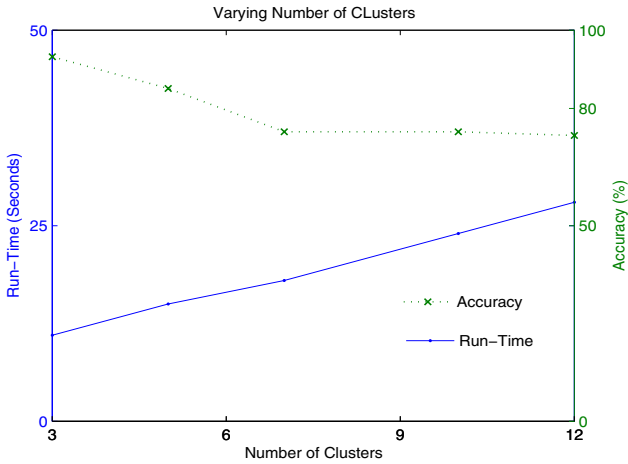


Fig. 3. Run-time and accuracy for different numbers of clusters

accuracy decreased a little when the number of documents increased. The reason was that more documents should cover much more terms, but we fixed the number of terms to 1100, which implicitly reduced the documents information and compromised the clustering quality.

Fig.3 shows the clustering run-time and accuracy against different numbers of clusters. The number of clusters varied from 3 to 12, while the number of terms was fixed to 500 and the number of documents to 1500. We can see that the *FW-KMeans* run-time also increased linearly with the number of clusters. The fixed number of terms resulted in the decreasing accuracy because it implicitly reduced the documents information in the raw text.

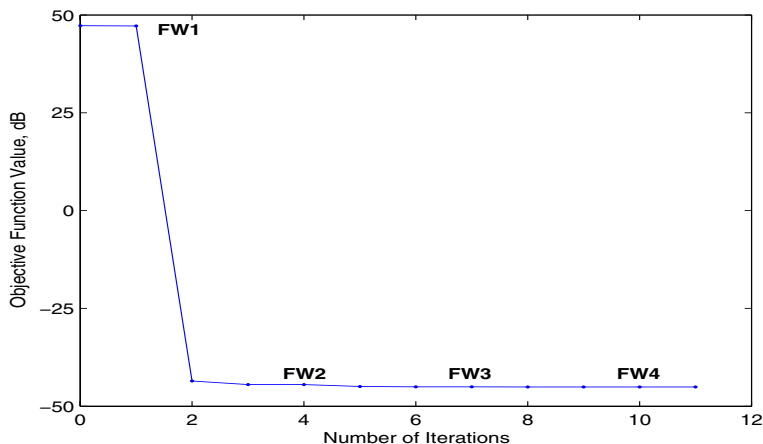


Fig. 4. Convergence curve of the *FW-KMeans* algorithm $dB = 10 \times \log(\text{value})$

Fig.4 shows a typical convergence curve of the *FW-KMeans* algorithm on the dataset (1500,500,7). The horizontal axis represents the number of iterations and the vertical axis represents the value of the objective function (1). Each point on the curve represents a partition generated by one iteration of the *k*-means clustering process. Starting from a set of initial centroids and a set of initial weights, the algorithm first converged after 2 iterations. A new set of weights *FW1* were computed. Using *FW1* as the initial weights and the current cluster centroids, the *k*-means process restarted again. We can see that the objective function had a significant drop after the new weights *FW1* were introduced. The *k*-means process converged again after 2 new iterations. Then, a set of new weights *FW2* were computed. This process continued until the local minimal value of the objective function was reached. The final set of weights *FW4* was obtained. We note that in each step, the weighted distance function was fixed since the weights for the features were fixed. Therefore the corresponding weighted distance function space was well-defined at each step. We expected that the smaller the value of objective function value, the closer the data points under the weighted distance function space. We remark that by using the similar arguments in Section 3, it can be shown that the above process is also convergent. For all the fourteen datasets, the iteration *h* is always less than 15, that is, our algorithm could converge quickly.

5 Conclusions and Future Work

In this paper we have shown the performance of the subspace clustering algorithm with feature weighting k-means. We analyzed this algorithm in terms of computational complexity, space consumption and convergency. A series of benchmark tests were conducted on the large real-world text datasets. The ex-

perimental results have shown that this subspace clustering method is efficient and scalable to cluster large and sparse text data in subspaces.

In the next step, we plan to integrate ontology, e.g. WordNet [15], as background knowledge to enhance our method in text clustering and mining. The ontology will sever several purposes in the clustering process, including data preprocessing, selection of initial cluster centers, determination of the number of clusters k , and interpretation of clustering results. We believe that ontology will provide premising solutions to the big problem in text mining: complex semantics.

References

1. A. Jain and R. Dubes, "Algorithms for clustering data," *Prentice-Hall*, 1988.
2. O. Etzioni and O. Zamir, "Web document clustering: a feasibility demonstration," *ACM SIGIR*, pp. 46–54, 1998.
3. Y. Zhao and G. Karypis, "Comparison of agglomerative and partitional document clustering algorithms," *Technical report #02-014, University of Minnesota*, 2002.
4. L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: a review," *SIGKDD Explorations*, vol. 6, no. 1, pp. 90–105, 2004.
5. R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," *In Proceedings of the 1998 ACM SIGMOD international conference on Man-agement of data*, pp. 94–105, 1998.
6. L. Jing, M. Ng, J. Xu, and Z. Huang, "Subspace clustering of text documents with feature weighting k-means algorithm," *PAKDD*, 2005.
7. M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," *Proc. Text ming workshop, KDD*, 2000.
8. J. Ghosh, "Scalable clustering methods for data mining," *Handbook of data mining*, 2003.
9. B. Ricardo and R. P.Berthier, "Modern information retrieval," *Addison Wesley*, 1999.
10. Y. Chan, K. Ching, K. Ng, and Z. Huang, "An optimization algorithm for clustering using weighted dissimilarity measures," *Pattern recognition*, vol. 37, no. 5, pp. 943–952, 2004.
11. P. Hague and P. Harris, "Sampling and statistics," *Kogan Page*, 1993.
12. J. Bezdek, "A convergence theorem for the fuzzy isodata clustering algorithms," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 2, pp. 1–8, 1980.
13. Y. Yang and J. Pedersen, "A comparative study on feature selection in text categorization," *ICML97*, pp. 412–420, 1997.
14. A. McCallum, "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering," 1996. [Online]. Available: <http://www.cs.cmu.edu/mccallum/bow>
15. C. Fellbaum, "Wordnet: an electronic lexical databases," *The MIT Press*, 1998.

State Transfer Graph: An Efficient Tool for Webview Maintenance

Yan Zhang¹ and Xiangdong Qin²

¹ Center for Information Science, Peking Univ., Beijing, 100871, China
zhy@cis.pku.edu.cn

² Science and Technology Department, Hebei Univ., Baoding, 071002, China
qinxd@mail.hbu.edu.cn

Abstract. When the web becomes more and more whirling with tremendous data, traditional maintenance approaches for web warehouses exhibit poor performance. In this paper, we first demonstrate the requests to webviews has continuity, and then we propose the *State Transfer Graph* (STG), an adaptive tool for webview maintenance. Using STG, we can produce many concrete maintenance methods, which will effectively service the dynamically changing web warehouses.

As a demonstration, we illustrate two particular methods named MEDI and VMF. The MEDI approach, which has three states for webviews, outperforms the previous approaches significantly when both of the web changes and the query requests are frequently. The VMF method, which has five states for webviews, is more powerful when the web becomes more and more dynamic. It improves query performance up to 59.4% over the minimum-update approach, when we enhance it with a global structure.

1 Introduction

1.1 Problem Statement

A webview is a materialized view usually defined by a query, which is generally related to a specified topic. Webviews enable users who are interested in specified topics to query a single data structure instead of issuing lots of queries over different data sources and structures [20]. Previous work [2][4][10] has presented lots of methods to make better arrangement for refreshing the materialized views. However, as the web is becoming more and more dynamic, webview maintenance is still a challenge, especially for those large scale web warehouses that have tremendous webviews and relatively limited refreshing resources. Traditionally there are two extreme approaches for refreshing the integrated views: the lazy-update approach [12] [13] does re-computation for each query request, while the eager-update approach [20] re-computes the views for each update request. However, both of them are optimized for some special cases. While the amount of data in web warehouses is increasing and the need to provide up-to-date results to the query load becomes more and more urgent, the time window available

for refreshing the web warehouses has been shrinking [14]. Therefore, these two approaches can not satisfy the users any longer.

So emerges this method: re-compute webviews only when necessary. For the update request, it delays the re-computation, remembers this update request and turns on a *flag*. When a query request arrives, it checks the status of the *flag*. If the *flag* is on, it re-computes the webview. If not, it directly uses the current value for the query. It is very easy to prove that this approach achieves the minimized updates, so it is called the minimum-update approach. This approach can be regarded as an improvement of the lazy-update approach. It works well when the data sources of the web warehouse (i.e., the real web information) are not very dynamic. However, it takes a longer time for query latency than the eager-update approach, and this problem becomes very serious when the web information changes rapidly (unfortunately, the real web is changing more and more rapidly). Therefore, we have to pursue more efficient techniques to keep up with the steps of the real web.

1.2 Our Solution and Contribution

Through analyzing the maintenance approaches in previous studies, we find a common characteristic: all of the approaches can be demonstrated by a group of transfers among some particular webview states. For example, in the eager-update approach, a view has only one state *Active*, which means the view is always up-to-date and can be used directly. In the lazy-update approach, a view has only one state *Sleeping*, which means the view does not respond to any updates but will be computed on the fly. For the minimum-update approach, it provides two states, *Active* and *Sleeping*, for the web views (see Section 3). This observation motivates us to explore and propose the *State Transfer Graph* (STG), a general paradigm to describe the behaviors of the webviews.

A *state* describes a specific status of a webview, while a *state transfer graph* is an abstraction model to describe how a webview moves among these *states*. Therefore, a concrete approach can be drawn from a given STG. According to our study, during a long period, query requests and update requests usually are continuous respectively, especially when the web becomes more active. This so called “the continuity phenomenon” makes our STG paradigm self-adaptive to the changing web. When the real web becomes more and more whirling with tremendous data and requests, we can introduce more states to simulate the webview behaviors. As a result, a more suitable STG is produced and the maintenance method derived from this STG will have a good performance.

As a demonstration, we present two particular methods named MEDI and VMF in this paper. There are *three* and *five* states for webviews in these two methods, respectively. We show that when the web is not very dynamic, the previous approaches can work fairly well. However, when the web changes rapidly, the MEDI approach outperforms the previous approaches significantly. If the web keeps its changing trend and the queries are also increasing expeditely, we have to apply the VMF method to the webviews.

The VMF approach demonstrates a better performance than all the previous approaches, according to our experimental results. For example, it can reduce the query latency up to 44.1%, compared with the minimum-update approach. Further, when we enhance the VMF with a global structure, we can achieve a query performance improvement up to 59.4%. In addition, this approach is scalable to the dynamic web environment, which means it can achieve much benefit no matter the web is active or not.

The rest of this paper is organized as follows. Section 2 gives a brief description of the background and related work. Section 3 introduces “*the continuity law*”, explains the motivation of proposing STG, and presents the MEDI approach as an example. Section 4 demonstrates VMF, a more powerful method derived from STG. Section 5 discusses the enhancement of VMF with a global structure. Section 6 evaluates all the approaches mentioned in this paper. Finally, we conclude this paper in Section 7.

2 Background and Related Work

Materialized view is an important and powerful technique in data warehouses. They allow users to query potentially terabytes of detail data in seconds, rewriting the queries by using kinds of materialized views. However, since it is meaningless to provide the stale data to users, the popularity of the materialized views necessitates efficient techniques for their maintenance, especially when the amount of data in a warehouse and the number of materialized views are rapidly increasing [2] [8][14][17][21]. Previous work usually uses an incremental approach or builds a global plan. For example, Gupta et al. illustrate when update sizes are small in relation to the sizes of source data, the incremental method is generally less expensive than recomputing the views from scratch [9]. Agrawal et al. propose a SWEEP algorithm for efficient incremental view maintenance at data warehouses [2]. Mistry et al. exploit common sub-expressions in their efficient maintenance plan [14]. Goldstein and Larson present a fast and scalable algorithm for determining whether part or all of a query can be computed from materialized views and describes how it can be incorporated in transformation-based optimizers [8]. Yi et al. try to achieve runtime self-maintenance with high probability by maintaining a dynamic top-k view [21].

The above work mainly focuses on the relational data model. As semi-structured data model is becoming dominative in integrating heterogeneous data sources, researchers begin to pursue powerful algorithms for semi-structured data. For example, Abiteboul et al. start from the graph-based data model OEM, develop an analytic cost model, and propose an algorithm which can produce a set of queries that compute the changes to the view based upon a change to the source [1]. Papakonstantinou and Vassalos present an algorithm to find rewriting queries for a given semistructured query and a set of semistructured views [16]. Cluet et al. believe XML will play an important role in the world of the web, and they present a view mechanism for their Xyleme system, addressing the problems of defining, storing, and using views in a web scale database [18]. How-

ever, these studies do not consider the difference between materialized views in traditional data warehouses and webviews in web warehouses. Traditional data warehouses usually know the changes of base data, while the data sources in a web environment usually do not propagate their changes to the information consumers. Therefore, a web warehouse has to employ a probing mechanism to detect the changes of base data [4].

Cho et al. synchronize an integrated database to improve its freshness by detecting the change frequency of the base data [3][5]. However, they focus on the web pages and do not consider the webviews, which consist of the information comes from multiple web pages.

3 The Continuity Law and the State Transfer Graph

Both query response time and system maintenance cost are important in a large-scale web warehouse. Therefore, we can formulate the webview refreshing problem as following: how to allocate the refreshing resource for each webview, so that we can achieve the minimized average query response time? Generally speaking, this problem is NP-hard, just as the multiple query optimization problem [19]. However, there are some effective approximate algorithms.

3.1 The Continuity Law

Let us start our analysis from the eager-update approach [20]. Our motivation is to reduce the number of re-computation, without significantly increasing the response time. Along with the exploding of web information, both the query requests and update requests in web warehouses become more and more frequent. If each update to the base data leads to an immediate refresh of webviews, the heavy maintenance burden will cause serious performance degradation. Through a long time observation, we find that a webview does not receive a constant attention during its lifespan. A webview is generally related to a specified topic, and users show special interest on a given topic only in some particular periods. Therefore, the query requests to a specific view are very frequent in these periods, often continuous, and so do the update requests. In other words, a webview usually receives continuous query requests and continuous update requests, as shown in Figure 1.

Previous studies also validate our observation of this continuity [6][7][11] [15]. Cho et al. demonstrate that each web page has its infant, expansion, and maturity stages [6]. The page popularity increases rapidly in its expansion stage, and this increase is much more sudden under the search-dominant model than under the random-surfer model. We believe the same thing happens to the webviews: at the infant stage, a webview is always silent, while after the expansion it receives more and more query requests, usually continuous. When studying the change frequency distribution, Ntoulas et al. observe that a significant fraction of web pages (around 50%) never changed at all during the course of their study, while

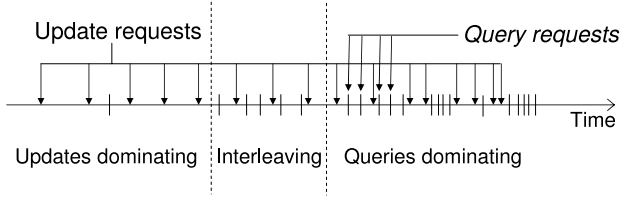


Fig. 1. During the lifespan of a webview, the update and query requests dominate in turn. More particularly, they tend to cluster together.

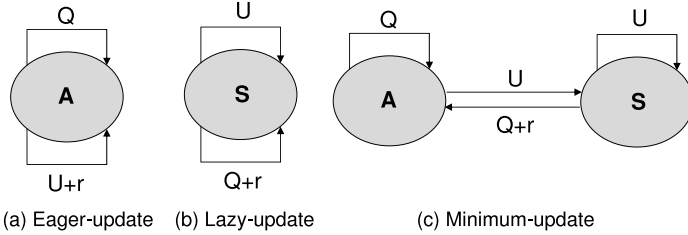


Fig. 2. The state transfer graphs

another quite large portion (15%) changed very frequently [15]. These observations also suggest that the query requests and update requests usually will be continuous respectively.

Based on the observation of this continuity, through analyzing the behaviours of many webviews as well as doing lots of experiments, we propose “*the continuity law*” on temporal locality for the requests to webviews and believe it has strong rationality and feasible operability: *the next request to a webview is quite possible to have the same type as the requests that occurred most frequently in the recent period*. Inspired by “the continuity law”, we can improve the previous maintenance methods. For example, there is no need to do an immediate re-computation for an update request, if we know this update request is followed by another update request. To make this idea more clear, we propose a term *state* for webviews and an effective tool *state transfer graph* to further describe the *states*.

3.2 State Transfer Graph

A *state* describes a specific status of a webview. Each state has its attributes and the activity is the main one. The activity of a webview increases when it receives queries and decreases when it receives updates. The current state for a webview describes its current activity, and reflects the ratio of the query requests over update requests.

A *state transfer graph* is an abstraction model to describe how a view moves among these *states*. It is a directed graph, in which the nodes are the *states*, and the directed edges represent the transfer conditions as well as the actions

that should be taken. We illustrate the state transfer graphs for the previous approaches, as shown in Figure 2. “**Q+r**” means when a view receives a query (**Q**) request, it should be re-computed.

In an eager-update approach, a view has one state *Active* (**A** in short), which means the view is always up-to-date and can be used directly. To keep the view always *Active*, any update to the base data will lead to an immediate refreshing of this view. In a lazy-update approach, a view has one state *Sleeping* (**S** in short), which means the view does not response to any updates at all and will only be evaluated on demand. For the minimum-update approach, there are two states for each view: *Active* and *Sleeping*. A view transfers between these two states.

3.3 The MEDI Approach

Although the minimum-update approach achieves the minimized maintenance cost, sometimes we have to endure a long query response time, e.g., a sleeping view must be refreshed before being used. The reason is, the two existing *states* in the minimum-update approach represent the two extreme conditions. To alleviate this long latency, we introduce a state **M** (stands for *Median*, which is a median of *Active* and *Sleeping*). A view in state **M** is up-to-date and can be used directly. Thus the query latency can be reduced partially. Figure 3 illustrates this approach, which is named as MEDI (it stands for **MEDI**an).

In the MEDI approach, when a view in state **A** receives an update request, it will be re-computed immediately, thereby avoiding the latency for the next query. If the next request happens to be a query, we will benefit from this pre-computation. However, the activity of this view will decrease, hence its state transfer to *M*.

The MEDI approach takes advantage of “*the continuity law*”. By introducing more states into a system, we can have more flexibility to trade off the maintenance cost and the query response time. Moreover, we will have more choices for the attributes of states. Actually, we can obtain much more improvement in terms of the overall system performance, if we make deeper analysis and propose more appropriate states, as shown in next Section.

4 VMF Approach: A More Powerful Method from STG

The state transfer graph (STG) is a general paradigm to describe the behaviors of the webviews. When web information becomes more and more whirling and fleeting, we can introduce more states for webviews to describe their volatility. As a result, we will get a more suitable STG and then derive a better maintenance algorithm. In this Section we develop an more efficient approach, VMF (View Maintenance Based on A Five-State Transfer Graph), which can be regarded as an upgraded version of the MEDI approach.

In the MEDI approach, we refresh the webviews for each update request, hence the views can be used directly when queries arrive. However, if the next

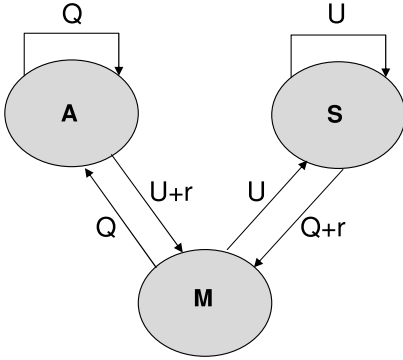


Fig. 3. The MEDI approach

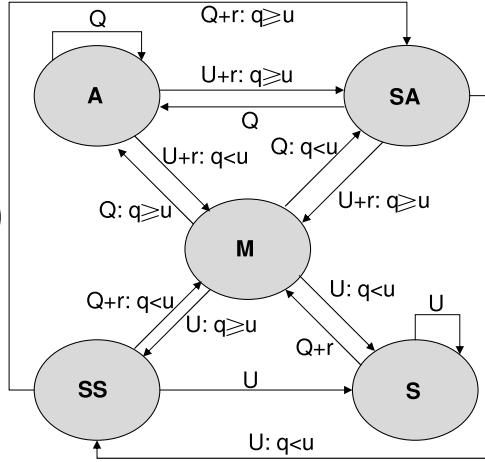


Fig. 4. The VMF approach

request is still an update, the previous re-computation will be a waste. This suggests we should predict the next requests in the request-sequence more variably. We believe more states can bring more flexibility. By splitting the state A and S, we introduce five states for a webview, A (*Active*), SA (*Semi-Active*), M (*Median*), SS (*Semi-Sleeping*) and S (*Sleeping*), instead of the three states in MEDI. Each state has its attributes, and the main one is the activity. When a view receiving a query request, its activity increases. In contrast, its activity decreases when receiving update requests. In general, the activities of state A, SA, M, SS and S decrease in turn. The first three states are up-to-date. The main difference between *Active* and *Semi-Active* is that a less active view usually can not transfer to *Active* state directly, unless it obtains enough activity. The main difference between *Sleeping* and *Semi-Sleeping* is that an *Sleeping* view will not transfer to *Semi-Active* state directly. It must go through the *Median* state. In VMF, our idea is to keep the active views *active* and the inactive views *sleeping*.

Figure 4 illustrates the idea of VMF. Q means *Query* request, U means *Update* request, r means *Re-computation* for the view, q and u represent the number of user query requests and the number of update requests respectively. $A \xrightarrow{U+r:q \geq u} SA$ means when an active view receives update requests, if $q \geq u$, the view needs to be re-computed, and then transfers from state A to state SA.

5 Algorithm Enhancement with a Global Structure

Although VMF achieves a good system performance, some important factors are still omitted in the basic algorithm. For example, the relationship between the re-computation cost and the query latency is very important. Moreover, we should pay more attention to the views with high query frequencies. Since our

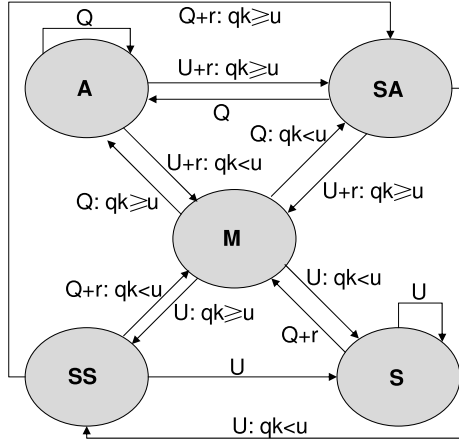


Fig. 5. The Enhanced-VMF approach

goal is to achieve the minimized query response time (under the constraint that the system capacity is a constant), we should allocate more system resources for the views whose re-computation costs are lower or whose importance are relatively higher. Thus we can achieve more benefit with same cost.

In Figure 5 we present the Enhanced-VMF approach, which illustrates this idea in detail. Here k actually is the abbreviation of k_i , which equals to $(q_i/TQ)/(r_i/TR)$. q_i is the number of the query requests to the view v_i , TQ is the total number of all query requests to all the views in the web warehouse, r_i is the re-computation cost of this view, TR is the total re-computation costs of all the webviews. By introducing the parameter k , we can keep the important webviews up-to-date, thereby saving the query response time.

The following example will demonstrate this enhancement clearly. In VMF, when a *Semi-Active* view receives an update request, it transfers to *Semi-Sleeping* immediately. However, this may not be true in the Enhanced-VMF approach. Whether it transfers to state *SS* is determined by the values of q , k and u . If the value of k multiplies q is larger than or equals to u , this view will be re-computed and keep state *SA*. Thus it can be used directly when receiving query requests.

In the VMF approach, each view works separately. However, when we introduce the parameter k_i for each view, all of them will collaborate together. Therefore the parameter k_i brings us a global structure.

6 Evaluation

6.1 Testbed Setup

We implemented the MEDI algorithm and the VMF algorithm, as well as the previous approaches in our testbed, a web warehouse mainly focused on the electronic commerce information. We collected 1.2 million web pages and stored

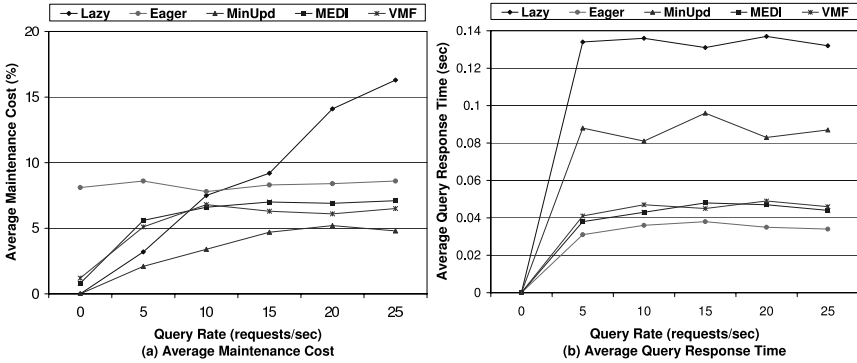


Fig. 6. Comparison of different approaches. The source data change frequency is 15 pages/second.

them locally. After that we constructed 10,000 materialized views based on these real data. The building process of the global structure was very complicated. We first detected the changes to the 1.2 million source web pages on the real web, and then chose the most frequently accessed pages and constructed a basic set of materialized views based on these pages. Secondly we generated lots of user query requests, simulating queries to the real websites. We recorded the queries and ranked them by query rates, added new materialized views that users wanted to know, and deleted the obsolete views and those nobody accessed. After one month, the web warehouse was finally built up and could be used for evaluation.

6.2 Overall Results

In the experiments, we make the locally stored base data change randomly to simulate the behavior of the real web. The warehouse changes accordingly.

We make the source data change at the rate of 15 pages per second, a reasonable number for electronic commercial web pages. By stochastically choosing queries from the real user requests set and varying the query frequencies to the web warehouse, we get the experimental result for the basic comparison.

In Figure 6, the unit of “average maintenance cost” means the percentage of the time being spent on the maintenance over the system processing time. From the Figure 6 we know, when query rate increases, MEDI and VMF will get a better performance gracefully. The eager-update approach achieves the minimized query response time, out of the four approaches. VMF and MEDI also do a good job. However, the lazy-update and minimum-update approaches take a bit longer. Compared with the minimum-update approach, VMF reduces the query latency up to 44.1%. In fact, when we get source data through the real web (not a LAN), we have to wait for much longer time to get the latest views, thereby the VMF approach will outperform much better than the minimum-update approach.

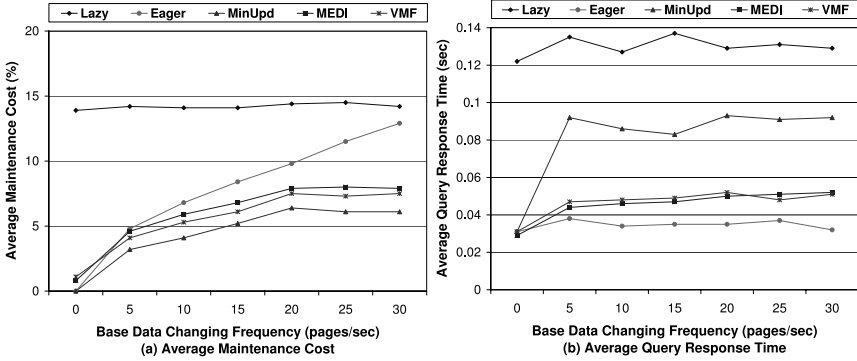


Fig. 7. Comparison of different approaches. The query frequency is 20 requests/second.

We then keep the query frequency unchanged (20 requests/sec) and vary the change frequency to the base data. In Figure 7, although eager-update approach achieves minimized query response time, its maintenance cost increases linearly along with the base data change frequency. VMF takes a little longer than the eager-update does, however, its maintenance cost increases along with the change frequency very slowly, eventually reaching a threshold. The minimum-update approach takes a quite longer time for the query response, though it costs the minimized system resource.

6.3 The More Dynamic Web

Now we make the “web” more dynamic. We accelerate the update rate to the base data, say, 30 pages per second. Keeping all the other parameters unchanged, we get the new measurement results. As shown in Figure 8, when the change frequency increases, the maintenance cost of materialized method increases evidently. In contrast, the cost of VMF increases very slightly.

Compared with the results in Figure 6, although the query latencies in MEDI and VMF are similar, the gap between their maintenance costs becomes larger. This indicates VMF is more suitable for a rapidly changing workload than MEDI.

6.4 Validating the Power of a Global Structure

The Enhanced-VMF has a global structure more than the basic VMF approach. How powerful is a global structure? Let us examine it. We know in the pure Enhanced-VMF approach, we must get the re-computation cost and the query rate for each view, which is very complicated. For simplicity, we classify the views into different groups instead of calculating the values for each view, and regard the views in the same group as equivalents.

We have two classification methods, using maintenance cost or query frequency. First, we categorize the views into two groups, according to their maintenance costs. The average maintenance cost of group 1 is 3.8 times over group

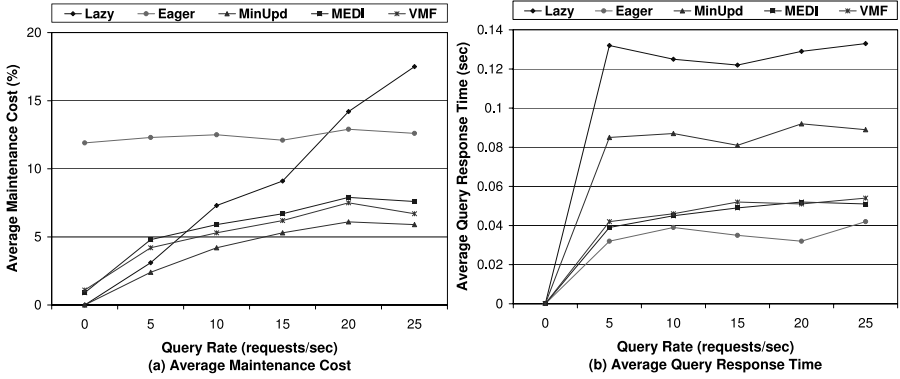


Fig. 8. Making the “web” more dynamic: improving the base data change frequency

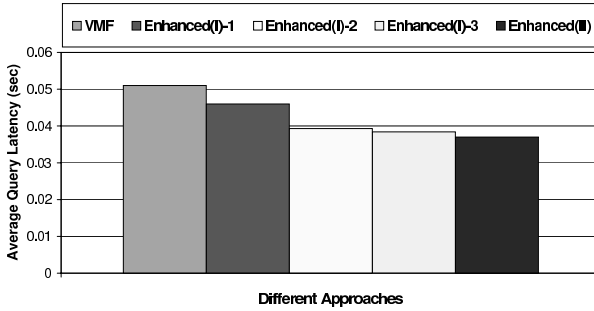


Fig. 9. Comparison of the VMF approach and the four enhanced approaches. In the Enhanced(I)-1, 2, 3 approaches, there are *two* groups of webviews. The Enhanced(II) approach has *four* groups for the webviews.

2. We access the warehouse 20 times per second. The change frequency to the base data is 30 pages/second. According to our experimental result, the maintenance costs are almost equal in the VMF and the Enhanced-VMF approach. However, the average query latency in the Enhanced-VMF approach is 0.046sec, which is less than the average query latency in the VMF approach, 0.051sec. The improvement is 9.8%. We change the average maintenance costs by choosing different views for the two groups and get the similar results.

Using the query frequency to classify webviews is more effective, compared with the maintenance cost. We vary the average query frequencies of the two groups in our experiment. When the ratio of the average query rate between the two groups increases, the Enhanced-VMOST approach achieves better result. As shown in Figure 9, Enhanced(I)-1, Enhanced(I)-2 and Enhanced(I)-3 represent the case that the ratio of the average query rate between the two groups is 2, 4 and 8, respectively. In Enhanced(II), we partition the webviews into four groups and the query frequency ratio among the four groups is 1:2:4:8. The average

query latency in Enhanced(II) is 0.037sec. It improves the query performance up to 27.5% over the VMF approach and 59.4% over the minimum-update approach.

As we see, more groups can help, however, the additional benefit is very slight. In other words, it is not worthwhile to employ a much more complex maintenance algorithm for the marginal additional benefit.

7 Conclusion

As the capacity of the web warehouses are rapidly increasing, the time available for maintaining webviews is dwindling, thereby making researchers pursue more efficient methods. In this paper, we first demonstrate the requests to webviews obey “*the continuity law*”, and then we propose *the State Transfer Graph (STG)*, a powerful and adaptive tool for webview maintenance. When the web becomes more dynamic, we can produce more efficient methods from the corresponding *STGs*. To the best of our knowledge, our work is the first study that employs a state-graph approach to solve the maintenance problems in web warehouses, or more generally, in data warehouses.

In this paper we present a three-state transfer graph method named MEDI and a five-state transfer graph method named VMF. We demonstrate both of them are very efficient, however, VMF outperforms MEDI when the web warehouse becomes more dynamic. Drawn from STG, there is lack of co-operation in VMF, which means each view works separately. To enhance such an approach, we establish a global structure by introducing a parameter k_i . As expected, this parameter enables the warehouse to keep the frequently-accessed views up-to-date, thereby reducing the query latency significantly.

References

1. S. Abiteboul, J. McHugh, M. Rys, V. Vassalos, and J. L. Wiener. Incremental maintenance for materialized views over semistructured data. In *Proceedings of VLDB'98*, Aug. 1998.
2. D. Agrawal, A. E. Abbadi, and A. Singh. Efficient view maintenance at data warehouses. In *Proceedings of SIGMOD*, pages 417–427, 1997.
3. J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *In Proceedings of SIGMOD*, 2000.
4. J. Cho and H. Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology*, 3(3), August 2003.
5. J. Cho and A. Ntoulas. Effective change detection using sampling. In *In Proceedings of 28th International Conference on Very Large Databases*, September 2002.
6. J. Cho and S. Roy. Impact of web search engines on page popularity. In *In Proceedings of the World-Wide Web Conference (WWW)*, May 2004.
7. G.K.Zipf. *Human Behavior and Principle of Least-Effort*. Addison-Wesley, Cambridge, MA, 1949.
8. J. Goldstein and P.-A. Larson. Optimizing queries using materialized views: a practical, scalable solution. In *In Proceedings of SIGMOD*, 2001.

9. A. Gupta, I. S. Mumick, and V. S. Subrahmanian. Maintaining views incrementally. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 157–166. ACM Press, 1993.
10. H. Gupta. Selection of views to materialize under maintenance cost constraint. In *Proceedings of International Conference on Database Theory*, pages 453–470, 1999.
11. L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. On the implications of zipf’s law for web caching. In *Proceedings of IEEE INFOCOM’99*, March 1999.
12. B. Ludscher, R. Himmeroder, G. Lausen, W. May, and C. Schleppehorst. Managing semistructured data with FLORID: A deductive object-oriented perspective. *Information Systems*, 23(8):589–613, 1998.
13. B. Ludscher, Y. Papakonstantinou, and P. Velikhov. A framework for navigation driven lazy mediators. In *Proceedings of WebDB’99*, 1999.
14. H. Mistry, P. Roy, S. Sudarshan, and K. Ramamritham. Materialized view selection and maintenance using multiquery optimization. In *In Proceedings of SIGMOD*, 2001.
15. A. Ntoulas, J. Cho, and C. Olston. What’s new on the web? the evolution of the web from a search engine perspective. In *In Proceedings of the World-Wide Web Conference (WWW)*, May, 2004.
16. Y. Papakonstantinou and V. Vassalos. Query rewriting for semistructured data. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 455–466. ACM Press, 1999.
17. N. Roussopoulos. Materialized views and data warehouses. *SIGMOD Record*, 27(1), March 1998.
18. S. Cluet, P. Veltri, and D. Vodislav. Views in a large scale xml repository. In *In Proceedings of 27th International Conference on Very Large Databases (VLDB2001)*, pages 271–280, September 2001.
19. T. Sellis and S. Ghosh. On the multiple-query optimization problem. *IEEE Transactions on Knowledge and Data Engineering*, 2(2):262–266, 1990.
20. L. Xyleme. A dynamic warehouse for xml data of the web. In *IEEE Data Engineering Bulletin*, 2001.
21. K. Yi, H. Yu, J. Yang, G. Xia, and Y. Chen. Efficient maintenance of materialized top-k views. In *Proceedings of 19th International Conference on Data Engineering*, March 2003.

An EJB-Based Very Large Scale Workflow System and Its Performance Measurement

Kwang-Hoon Kim and Hyong-Jin Ahn

Collaboration Technology Research Lab,
Department of Computer Science, Kyonggi University
San 94-6 Yuiddong Youngtongku Suwonsi Kyonggido, South Korea 442-760
{kwang, hjahn}@kyonggi.ac.kr
[Http://ctrl.kyonggi.ac.kr](http://ctrl.kyonggi.ac.kr)

Abstract. The design and implementation of a workflow management system is typically a large and complex task. Decisions need to be made about the hardware and software platforms, data structures, algorithms, and the network interconnection of various modules utilized by various users and administrators. These decisions are further complicated by requirements such as flexibility, robustness, modifiability, availability, usability, and scalability. The scalability among them is the most impeccable issue, in recent, as the sizes of workflow systems and its applications are becoming much larger. Also, the organizations are recognizing that the de facto standard of workflow architectures, the activity-oriented workflow architecture that is the joint-flow reference model proposed by OMG, and its off-the-shelf and more prevalent in the near future. In this paper¹, we propose a solution for the scalability issue - The e-Chautauqua VLSW architecture and system.

For the realization of scalable workflow systems, we propose an architecture that is called workcase-oriented workflow enactment architecture, and implement an EJB-based workflow management system preserving the property of the architecture. We analytically and experimentally demonstrate that the architecture gives the best performance for the very large number of workcases (instances of workflow procedures). That is, our analytic results showed that (a) software architecture is as important as hardware architecture for VLSW performance enhancement, and (b) the proposed architecture is much more scalable. Based upon the analytic results, we have conceived the workcase-oriented workflow architecture, and experimentally demonstrated that our system is much better scalable and gives the reasonable performance for the very large workflow applications.

Keywords: Degree of Workflow Complexity, Taxonomy of Very Large-Scale Workflow Architectures, Workcase-Oriented Workflow Architecture, Activity-Oriented Workflow Architecture, Very Large-Scale Workflow Management System.

¹ The research was supported by the research fund of KOSEF (Korea Science and Engineering Foundation), No. R05-2002-000-01431-0.

1 Introduction

In fact, workflow is becoming one of the hottest technologies in the enterprize information technology arena, which means that workflow systems have rapidly been adopted by many organizations. The primary reason why workflow systems are so hot-issued comes from the belief that they help large organizations to improve dramatically the way they operate, and their effectiveness has also come under our observation in numerous deployments. But, many of the workflow products have been experienced being ineffective for serious workflow applications because they generally lack necessities for modeling capability, scripting tools, dynamic change supports, and many other advanced features of the workflow systems. Particularly, many successful large workflow customers found out that as they tried to scale up from pilot test mode to enterprize-wide mode, there were severe symptoms of inflexibility and non-scalability[1]. Therefore, the recent requirements on the workflow systems have been set up and characterized by the challenge - Performance and Scalability.

In the very large scale workflow domain, the degree of complexity that affects the system's scalability and performance can be determined by the factors - workflow engagement factor, workflow structure factor and workflow instantiation factor. The workflow engagement factor has to do with how large the organization is - How many workflow models including actors, roles and application programs does the system have to handle? The workflow structure factor has to do with how large and complex a single workflow model is - How many activities are associated within a single workflow model or collaborative workflows ? and how complex their control/data flow relationships are? - And, the workflow instantiation factor has to do with how large the number of instances coming out of workflow models is - How many instances do the workflow models produce? We, in this paper, try to estimate and measure the facts how well our architecture and system is coping with these factors, architecturally as well as systematically.

In terms of the more practical point of view, as organizations transfer more and more of their paperwork to electronic systems, the workflow system must grow to accommodate much larger numbers of users, work items (workcases), information items, and complex workflow applications, in general. So, the goal is to develop a workflow architecture and system that not only be useful for workflow applications as small as 50 or 100 simultaneous workcases of a workflow procedure, but also be scalable up to 500,000 or 1 million workcases without any seriously performable degradation. As a consequence of this, the architecture and system is able to provide acceptable performance for much larger and much more complex workflow applications, too. We name it 'e-Chautauqua² Workflow Management System'. The system is developed by the EJB-based framework

² The origin in the Webster's dictionary: any of various traveling shows and local assemblies that flourished in the United States in the late 19th and early 20th centuries, that provided popular education combined with entertainment in the form of lectures, concerts, and plays, and that were modeled after activities at the Chautauqua Institution of western New York

approach, and it is fully deployable and operable now on the internet-based computing environment.

Through the following four more sections, we present descriptions of the degree of workflow complexity for scoping our paper's goal. In the next section, we propose a large scale workflow architecture and illustrate its mathematical estimations of performance by comparing with other conventional workflow architectures. In consequent sections, we describe about details of the e-Chautauqua workflow management system and its runtime execution experience and performance measures, and introduce several related works that our research group and the workflow literature have done for issues of the large scale workflow architectures and their performances. Finally, we summarize what we have done, the results of which workflow architectures are adequate and adaptive for very large-scale workflow applications, and what the future works and further research issues are.

2 Related Works

Nowadays, there exist hundreds of commercial workflow management products. The genesis of workflow management software was in automating document-driven workflow procedures such as officeTalk-D[5], etc.. Some of the early products were extensions to the document imaging and management software. The emphases of those workflow products have been on routing, sharing, application launching, and cooperating activities of workflows. Meanwhile, the current products are pursuing the issues of transactional workflows and large scale workflows. Especially, the feature of supporting the '*Large-scale Workflows*' is still hot-issued in the literature, because the architectures of almost all workflow products are belonging to a family of the client-server architecture that suffers from performance and scalability problems in the situation where it is faced with large scale workflow applications. More scalable architectures need to be supported for the large scale workflows.

The scalability issue is the problem that we are concerning about in this paper. As mentioned before, many workflow management systems have been developed and described in the workflow literature; we cannot describe them all here. Almost all commercialized products, such as BizFlow, InConcert, Staffware, FloWare, Notes, ActionWKF, FlowWorks and FlowMark workflow management systems, have adopted a certain type of advanced workflow architectures employing a sort of the EJB enterprize middleware platform. Particularly, JBoss jBPM is a typical example of the new paradigm recently happening in the literature. It is pursuing a flexible, extensible workflow management system as the open source enterprize middleware platform. JBoss jBPM can be used in the simplest environment like an ant task and scale up to a clustered J2EE application. At first, we have developed the e-Chautauqua workflow system so as to be operable on the Weblogic EJB web application server framework, and now it is deployed on the JBoss EJB enterprize middleware platform.

3 The EJB-Based Very Large Scale Workflow System

We have developed a workflow management system that aims for very large scale workflows applications, and it is named e-Chautauqua workflow management system. e-Chautauqua is based on the workcase-oriented workflow architecture, and especially we have implemented it by the Enterprize Java Beans framework approach. In this section, we describe e-Chautauqua's overall system architecture and its workflow message queue mechanism providing asynchronous communication channels among the agents. Also, in order to show that e-Chatauqua might be reasonably acceptable as a very large scale workflow management system, we gives two experimental observations on e-Chautauqua - One is an operational example by capturing the runtime client's screens, and the other is a performance evaluation result by measuring the response times for each of workcase objects to be created by the engine.

3.1 A Workflow Architecture Design Guidance Through the Analytic Performance Analysis Results

In [10], we developed five architectural performance analytic models by using the layered queueing model, and mathematically analyzed them by the method of layers methodology. One of the models is of the passive workflow architecture (denoted by PM /w 18 Workflow Enactment Processes in Fig. 1), two are of the class active workflow architectures (CAM /w 2 CPUs and CAM /w 18 CPUs), and the other two models are of the instance active workflow architectures (IAM /w 2 CPUs and IAM /w 18 CPUs). See [10] for more detail on the larger view of the analyses. In here, we present just the results of the architectural performance estimations as shown in Fig. 1. The results shows that the average response time is much larger in the passive workflow architecture (PM) than in the class-active workflow architectures (CAM) as well as in the instance-active workflow architecture (IAM), even though the passive workflow architecture increases not only the number of workflow engine servers (processes) (PM w/ 18 WEPs) but also the number of hardware processors (PM w/18 CPUs). This means that the instance-active workflow architecture has a higher performance in a more distributed environment as the workflow scales up.

Note that, as seen in the graphs, the performance differences between the passive workflow architectural category and the active workflow architectural category (the class-active and the instance-active workflow architectural style) are very large. The active workflow architectural categories' performance is far superior to the passive workflow architectural categories'. The outcomes of this performance analysis work might be predicted based on the conceptual intuition. But, the derived comparisons between the first instance-active workflow architecture (IAM w/ 2 CPUs) and the second instance-active workflow architecture (IAM w/ 18 CPUs) makes us a bit surprised, because the first's performance is better than the second's, which is not what we expected from the conceptual intuition. Therefore, it might be true that it will be a burden if the number of executable processes in the system increases without limit. That is, in the

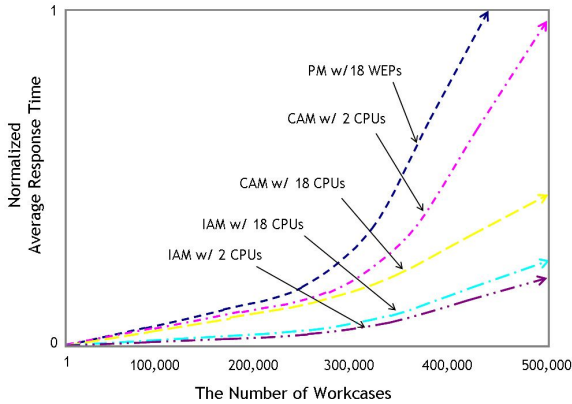


Fig. 1. The Analytic Performance Evaluation Result

instance-active workflow architecture, the number of executable processes or objects, which are proportionally related to the number of workcases, must be inflicting heavy effects on its performance. As a result, based on these estimations we are able to reach the following guidance in conceiving a very large scale workflow architecture:

- **Rule 1.** *The instance active workflow architectural style is more appropriated for very large scale workflow systems.*
- **Rule 2.** *The number of executable processes or objects composing a workflow engine should be maintained as small as possible.*
- **Rule 3.** *Both of the above rules should be simultaneously satisfied, but Rule 1 takes precedence over Rule 2 if it is not.*

3.2 The Workcase-Oriented Workflow Architecture of e-Chautauqua

In this section we try to derive a conceptual workflow architecture from the guidance. At first, in order to keep the Rule 1, it is necessary to understand which entity of a workflow model is transformed or embodied into active components (such as a process, object or thread) of the architecture. That is, workflow procedures, activities, workcases, roles, actors must be the typical entities of a workflow model, and the instances that are instantiated from any entity among them can be embodied into active components in the instance active workflow architecture. It should be reasonable if we choose the activity entity or the workcase entity as a subject of embodiment in the architecture. So, with respect to the Rule 1, it is possible to derive two candidates as a very large scale workflow architecture - One is an activity-oriented instance active workflow architectural style, the other is a workcase-oriented instance active workflow architectural style. Based on the Rule 2, we have to choose the workcase-oriented instance active workflow architectural style as a very large scale workflow architecture, because we obviously know that the number of executable processes or objects

maintained in the workcase-oriented architecture is much smaller than in the activity-oriented architecture. Finally, we reach the decision that the workcase-oriented instance active workflow architectural style, which is abbreviated to the workcase-oriented workflow architectural style, is the proper conceptual architecture for very large scale workflow management systems.

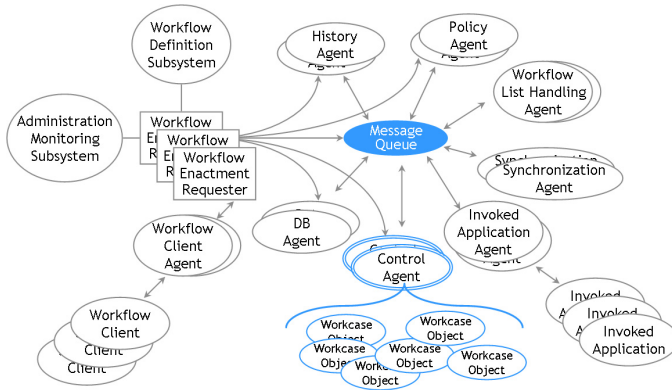


Fig. 2. The Workcase-oriented Workflow Architecture

In Fig. 2, we present a possible conceptual architecture preserving the workcase-oriented workflow architectural style. There need to be several different kinds of agents in the architecture, and however the most essential agent is the control agent taking in charge of the control flow functionality that is the primary role of a workflow engine. So, the core work in deriving a workflow architecture is on how to configure and organize the inside of the control agent. Conclusively, in the workcase-oriented workflow architecture, a set of workcase objects fulfill the control agent’s functionality, and a workcase object can be realized as either a work item (token) or an instance of workflow procedure in a workflow system. Therefore, it is possible to consider two kinds of workcase objects - One is the case that a token becomes an active component and takes in charge of activity enactment control by itself, and the other is the case that an instance of workflow procedure becomes an active component whenever a workcase is initiated. After completing an individual workcase, the active components (workcase objects) corresponding to the workcase are automatically destroyed. We believe that the workcase-oriented instance-active architecture should be very appropriate for maximizing the distribution capability and the resources’ usage efficiency in performing the enactment scheduling and event-handling tasks. Note that almost all of the current workflow systems that are developed based on the OMG Joint-flow’s workflow architectural style are categorized in the activity-oriented instance active workflow architectural style, and so their control agents are composed of a set of activity objects. In the next section, we experimentally show that the workflow management system embedding the workcase-oriented

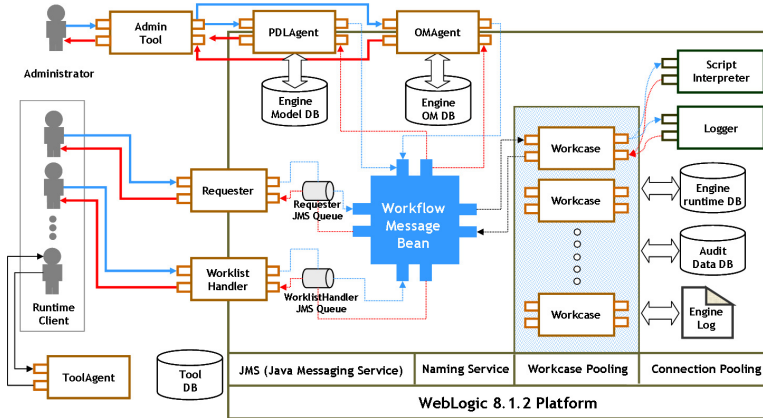


Fig. 3. The Overall System Architecture of e-Chautauqua

workflow architecture is able to afford feasible performances in very large scale workflow workloads.

3.3 The Overall System Architecture of e-Chautauqua

The functional descriptions of e-Chautauqua’s architectural components and the ways they cooperate with each other are briefly explained in this section. Basically, e-Chautauqua is fully implemented so as to provide almost all services and facilities specified in the international workflow standards from the interfaces 1 through 5 announced by WfMC. For examples, in terms of the workflow definition language, it completely handles WPDML as well as XPDL, and also it is able to support the Wf-XML 2.0 interoperability interface, which has recently released by WfMC, for interacting with other heterogeneous workflow systems. Our focus is on the workflow enactment component (the so-called workflow engine).

The overall system architecture of e-Chautauqua as shown in Fig. 3 consists of several agent-based components, such as PDL (process definition language) Agent, OM (organization management) Agent, Administration Agent, Tool Agent, requester, worklist handler, script interpreter, logger, workflow Message Beans using JMS message queue, and Enactment & Control Agent consisting of workcase objects registered in the workcase pool. These all components are deployed on an Enterprise Java Beans middleware platform³ and they asynchronously communicate with each other through a JMS-based message queue mechanism named the workflow message beans.

³ Note that we used the Weblogic 8.1.2 package, but it won’t be dependent on a specific EJB package. Recently we have successfully deployed our system on JBoss enterprise middleware platform, too.

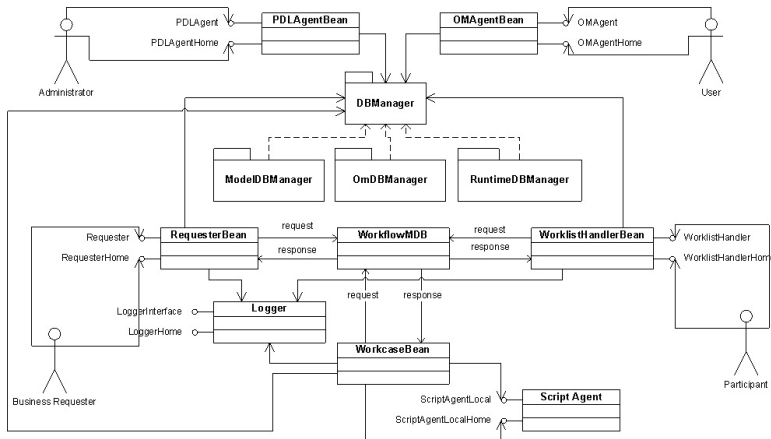


Fig. 4. e-Chautauqua Engine’s Class Diagram

The core component that distinguishes e-Chautauqua from other workflow engines is the workcase object component. That is, as stated in the conceptual architecture section, almost all conventional workflow engines are based on the activity-oriented workflow architectural style, but on the other hand e-Chautauqua’s engine is based on the workcase-oriented workflow architectural style. In the activity-oriented workflow engines, the control flow management functionality is performed through interactions among activity objects, because the core component is the activity object component. In contrast to this, the control flow management is done by the workcase objects in the workcase-oriented workflow engine (e-Chautauqua), and so the activity precedence information is stored to the inside of each workcase object as data. Conclusively, the most valuable benefit that we expect from the workcase object component is that the control flow management functionality can be done by much simpler mechanism, and also it can be efficiently done with much smaller amount of computing resources, because the workcase-oriented workflow engine is able to dramatically decrease the number of objects resided and managed in the system. These facts mean also that the workcase-oriented workflow engine like e-Chautauqua must be more appropriate and durable for the very large scale workflows, and it must give satisfaction in the higher degree of workflow instantiation complexity. Fig. 4 is to present the class diagram of e-Chautauqua engine as an implementation result of the system architecture of Fig. 3. The components in the system architecture of e-Chautauqua are implemented as workcasebean class, worklisthandlerbean class, requestbean class, and so on, respectively.

Let us briefly describe the remains of the components. The workflow process (procedure) builder and simulator are responsible for describing, specifying, and simulating workflow procedures through the graphical user interfaces. The graphical description of a workflow procedure can be represented in a textual form through the workflow procedure description language (WPDL) or the XML

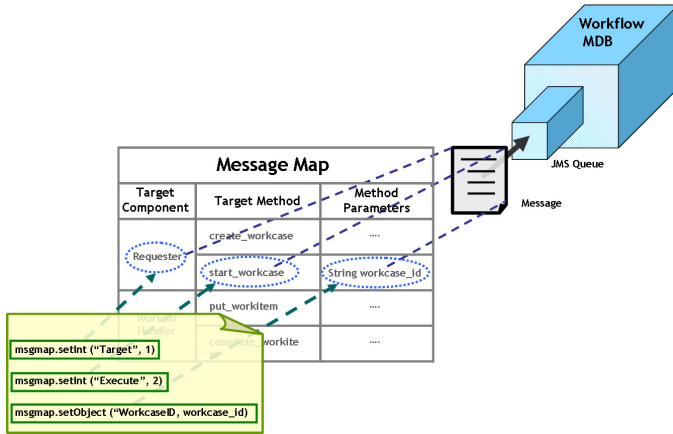


Fig. 5. e-Chautauqua's Workflow Message Queue Mechanism

workflow procedure description language (XPDL). The workflow administration and monitoring services are used for managing the administrative tasks, like initiating workcases, performing dynamic changes of data-flow and control-flow in a workcase, terminating or holding tasks of a workcase, monitoring the execution statuses of tasks associated with a workcase, and serving other administrative and monitoring requests. The services are operable on a graphical user interface, too. The script interpreter performs the definition of transition conditions in buildtime and their processing during runtime.

Finally, these all components of e-Chautauqua communicate with each other through the workflow message bean. The workflow message queue mechanism uses a message-driven bean for realizing the reliable message transmission, which is based upon the JMS (Java Message Server) queue as an asynchronous message communication channel. An example using the workflow message queue mechanism is illustrated in Fig. 5. That is, a component sending a message composes the message in a certain format and transmits it to a JMS queue, and then the JMS queue looks up the message map to find out the target component and its method and parameters by interpreting the received message from the sending component, and finally the JMS queue invokes the method for passing the parameters' values or for returning the execution results. Fig. 5 shows that the sending component is a runtime client trying to request starting a workcase, and the target component is the requester that takes in charge of the responsibility for starting the workcase.

3.4 Operational Example: e-Chautauqua Runtime Client

In this section, we present an operational example of e-Chautauqua by showing captured screens of the runtime client user interface and its related functionalities. In Fig. 6, The first screen from the lowest layer in the overlapped screens is the runtime client's user interface having a worklist folder, workflow (or process)

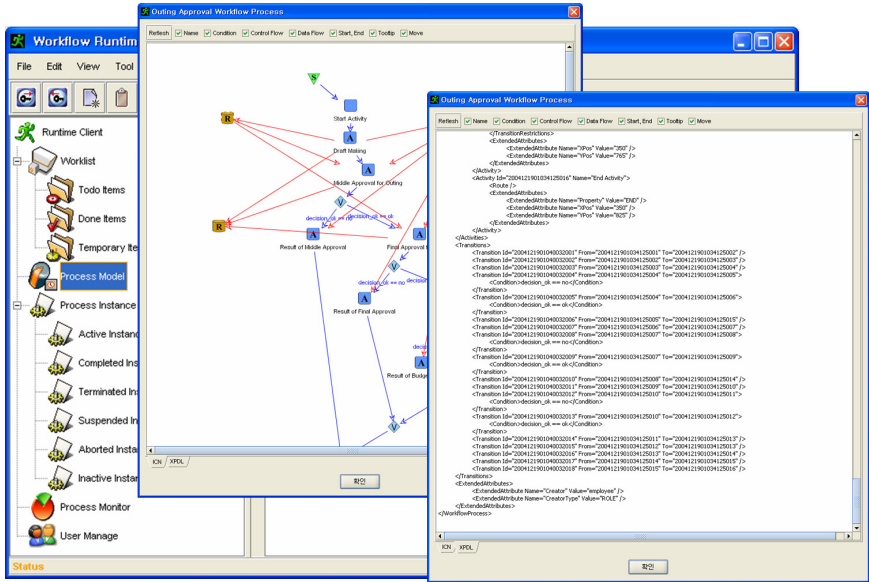


Fig. 6. e-Chautauqua’s Runtime Client

model folder, running workcases (instances) folder, and a workflow monitoring folder. The second screen is showing a workflow model graphically represented by the information control net, and on the top of the screen there are several toggle switches used for alternatively showing the workflow model information such as activity’s name, transition informations, control flow, data flow and so on. The last third screen shows the workflow model of the second screen represented in XPDL.

4 e-Chautauqua’s Performance Measurement

We have observed e-Chautauqua’s runtime executions for practically measuring its performance and evaluating whether e-Chautauqua is affordable for the higher degree of the workflow instantiation complexity as a very large scale workflow management system. In general, it is very hard to measure the elapsed time or the processing time of a workcase in a workflow system, because it’s not easy to define or predict the processing time of each activity in the workflow model. So, we measure the response time for e-Chautauqua to create a workcase object after being requested a start workcase request to the requester component. In order to conduct the performance measurement work, we built an automatic generator of the start workcase request messages. Fig. 7 shows the result of the performance measurement after measuring e-Chautauqua’s response times for more than one million workcases. As you see the central tendency graph in Fig. 7, the tendency line reveals a shape of exponential function from the point where

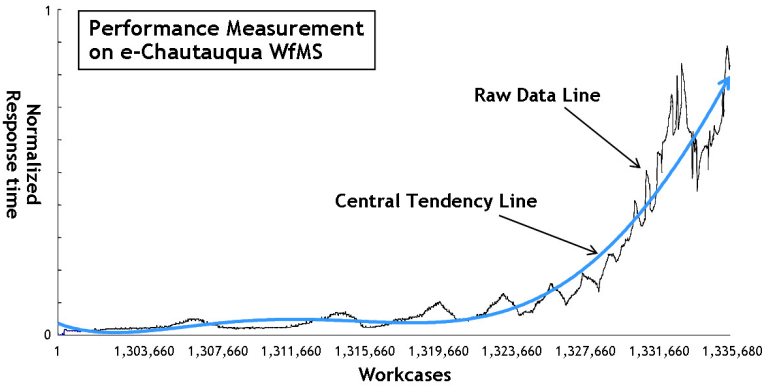


Fig. 7. e-Chautauqua's Performance Measurement Result

the number of workcases in the system starts passing the point of one million and three hundreds thousands workcases.

Based upon the performance measurement result, we have to make a decision if e-Chautauqua guarantees the acceptable performance for the very large scale workflow applications. We would say that the maximum number of workcases in the system should be large enough to accept the decision, because we deployed e-Chautauqua on the weblogic middleware platform running on a very simple hardware configuration. If we deploy the system on a more powerful hardware configuration like a clustering platform, then we can surely expect that the performance measurement result is much more reasonable for the very large scale workflow environment.

5 Conclusions

In this paper, we have newly defined the degree of workflow complexity in order to reasonably scope the very large scale workflow applications. Also, We have investigated performances of various alternatives of workflow architectures for very large scale workflow systems, especially which are satisfied with the higher degree of the workflow instantiation complexity. Based upon the results of our mathematical analysis, we analytically proved that the instance active workflow architectural style is much more scalable than any other styles of workflow architectures such as the server-client workflow architectural style, and the class active workflow architectural style. By considering the analytical proof, we have proposed the workcase-oriented workflow architecture for the very large scale workflow applications, and described the implementation details of e-Chautauqua workflow management system embedding the workcase-oriented workflow architecture. Finally, we have presented the performance measurement result of the e-Chautauqua workflow management system, and reached the decision that the architecture and system should be affordable for the very large scale workflow

requirements. Based upon the results of this paper, we can further dig our research into the world of Grid-based workflow architectures or/and P2P-based workflow architectures for exploring more advanced workflow architectures and systems targeting at the very large scale workflow applications.

References

1. Alonso, D., et.al., "Failure Handling in Large Scale Workflow Management Systems," IBM Research Report RJ9913, Nov. 1994
2. Alonso, D., et.al., "Exotica/FMQM: A Persistent Message Based Architecture for Distributed Workflow Management," Proceedings of the IFIPS Working Conference on Information Systems for Decentralized Organizations, Aug. 1995
3. K. Mani Chandy and Doug Neuse, "Linearizer: A Heuristic Algorithm for Queuing Network Models of Computing Systems", Communication of the ACM, Vol. 25, No. 2, Feb. 1982
4. Ellis, C. A. and Morris, P., "The Information Control Nets Model" Performance Evaluation Review, Vol. 8, No. 3, Nov. 1979
5. Ellis, C., "Team Automata," Proceedings of the ACM Group'97 Conference, Nov. 1997
6. Ellis, C. and Maltzahn, C., "Chautauqua: A Flexible Workflow System," Proceedings of the 30th HICSS Conference, January, 1997
7. Ellis, C. and Kim, K., "A Framework and Taxonomy for Workflow Architectures", The Fourth International Conference on Design for Cooperative Systems, May 2000
8. Jablonski, S. and Bussler, C., "MOBILE: A Modular Workflow Model and Architecture," University of Erlangen Internal Report, 1995
9. Kim, K. "Practical Experiences and Requirements on Workflow", Lecture Notes in Computer Science, Collaborative Technology for Collaboration Applications, Vol. 1364, pp. 145-160, Sept. 1998
10. Kim, K. and Ellis, C., "Performance Analytic Models and Analyses for Workflow Architectures", Information Systems Frontiers: A Journal, Vol. 3, No. 3, pp. 339-355, Aug. 2001

Deploying π -Calculus Technology in Inter-organizational Process

Memon Abdul Ghafoor, Jianwei Yin, Jinxiang Dong, and Maree Mujeeb-u-Rehman

College of Computer Science and technology, Zhejiang University, 310027, Hangzhou, China
ghafoorgem@yahoo.com, {zjuyjw, djx}@cs.zju.edu.cn,
mrmaree@yahoo.co.uk

Abstract. Seeing a great need of strong connectivity automation system in between different organizations for the promotion of the best services based upon reliable intermediate source, this paper presents the implementation cooperation of different services in the inter-organizational process locally and globally, that can play important role in sharing services required by many organizational platforms. We discuss how the Inter-organizational π -calculus technology can express organization communications within organizational distributed environment, through an encoding of the inter-organizational π -calculus technology, an enriched system that explicitly represents Names which are known universally but always refer to organizational information. Our presented translation replaces organization-to-organization communication shared with different organizations; we prove that this preserves and reflects process behavior, meets essential requirements for inter-organizational processes system based on an internet service daemon, and also investigate some limitations of the encoding.

Keywords: Inter-organizational π -calculus Technology, Business Process Management, Channels, Levels.

1 Introduction

Current trend in inter-organizational process system is flowing towards the inter-organizational business process where different organizations that are using their individual resources to achieve a common goal by modeling and executing inter-organizational workflow [1,2,3,4]. Inter-organizational workflow process system, however, raises numerous open research problems, such as how to model these workflows and the virtual enterprise [5,6], how to provide for the adequate service infrastructure to resolve heterogeneity such as its use in different organizational management systems. Inter-organizational workflow process system is essentially a set of loosely coupled workflow processes [7], typically where there are many business partners which are involved in one 'Inter-organizational' workflow process. Each of the partners has its own 'intra-organizational' workflow process. The inter-organizational workflow process system consists of intra-organizational workflows processes and an interaction structure. The examined example shown in Fig.1 where activities of participating organizations are counted as a set of intra-process provided

by the organization internally. The intra-process of an organization can provide a process service interface for external organization to participate in an inter-organizational process. The inter-organizational process uses as host-host and host-net processes to link a specified service. Each process service points to or is provided by the intra-process of an organization and the intra-process can be changed dynamically by adapting business process environment. Evolutions are always emerging from changes. For inter-organizational process, changes are actions to manipulate or operate the inter-organizational process and then the evolutions of dynamic inter-organizational processes are embodied in the effects of the actions. Therefore, reasoning about and analyzing the evolutions of dynamic inter-organizational processes is in fact to reason about actions and their effects. However, traditional formal methods can only describe constraints over actions instead of describing actions directly. Contrarily, the π -calculus can not only describe actions but also reason about actions. Thus, the π -calculus provides us a uniform approach to specify dynamic inter-organizational processes and reasons about the evolutions of dynamic inter-organizational processes.

This paper proposes an inter-organizational process system with inter-organizational π -calculus (*IO π -calculus*) technology techniques with primitives for location, migration and plain π -calculus style *channel* communication. The cooperation mechanism between organizations in distributed service process paradigm mechanism based on service utilization in a client/server infrastructure*.

In rest of this paper we introduce Inter-organizational Process Requirements in section 2. In section 3, Inter-organizational Process Advantages with π -calculus. In section 4, π -calculus syntax, scope and areas, and type systems. In section 5, proposed *IO π -calculus* technology. In section 6, π -calculus within organization with different aspects. In section 7, asynchronous π -calculus as a target π -calculus. In section 8, Compositional encoding of organizational π -term into the π -calculus. In section 9, Encoding correctness for organizational π -process. In section 10, Encoding of the Internet Server Daemon and Conclusion in section 11 respectively.

Related Work

Recently there is an increased interest in formal modeling of inter-organizational business processes. Several formalized languages for the specification of inter-organizational workflows have been proposed, each of them having different origins and pursuing different goals for dealing with the unique characteristics of inter-organizational process system. The two main formal tools used to model such workflow processes are Petri nets [8] based on bipartite graphs and process algebra [9,10] based on a textual description. In both areas there is an impressive accumulation of knowledge. Many notions developed for Petri nets can be translated to process algebra and vice versa. In π -calculus, the first step was taken by Milner [11] in 1991 who introduced an improvement of the π -calculus called polyadic π -calculus where channels were allowed to carry tuples of messages. Polyadicity supports concept of “sorting” and polyadic π -calculus maintaining the type discipline forces channels. Most

* This work is supported by National High Technology Project, China (Grant No.2003AA411021).

researchers focused about the π -calculus on variants of the language and proof techniques. Some researchers discuss the use of the π -calculus driven by the need to apply the π -calculus to practice. The first extension of Milner's system has been undertaken by Pierce and Sangiorgi, they distinguished between input-only, output-only, and input-output Channels. This extension naturally leads to recursive types with subtyping [12]. The use of PICT allows automatically generated applications from specifications used to investigate certain behavioral properties [13]. Vivas and Dam have studied the effect of these on the higher-order π -calculus and given an encoding into the π -calculus [14]. However, their encoding relies on blocking being carried out explicitly on individual *names*, and both complex and indirect. Cardelli, Ghelli and Gordon take a different approach to limiting communication with their notion of *name* groups [15]. In Inter-organizational distributed process systems, such as the Internet, sensitive administrative domains have to be protected against malicious agents, and tools are required for the analysis different properties. The distribution is one dimensional: in contrast with Distributed Join-calculus [16] or Mobile Ambients [17], locations do not contain sub-locations [18]. As in Mobile Ambients, communication is purely local; only co-located processes can communicate. Contrary to Mobile Ambients and Distributed join-calculus, mobility is weak in the sense that migrate code instead of computations.

Our presented inter-organizational process system with *IO π -calculus* technology techniques provides a simple but powerful framework to model fundamental features of distributed computations. Our focus is distributed π -calculus [19] which is based on the polyadic asynchronous π -calculus involving explicit and simple notions of locality and migration.

2 Inter-organizational Process Requirements

A typical inter-organizational process is enacted by a network of communicating entities exchanging information, with their actions taking place concurrently. The behavior of concurrent systems inherently non-deterministic as it cannot be predicted for the occurrence order atomic actions of the participating processes. This non-determinism may be unwanted, but a process modeling language must be able to capture it so that it can be avoided. Organizations today usually use products and services of other organization to deliver their products, or services, to clients, and such collaboration makes it necessary to integrate processes of the involved parties. At the same time processes can be important assets of organization that it does not want to reveal to the public or to competitors. To be useful, inter-organizational process modeling language must therefore be able to discriminate between internal, unobservable actions and those can and may be observed from the outside. The process model should then make it possible to verify that the internal behavior of the service satisfaction. In short, inter-organizational process must deal with communication, concurrency, constraints, differentiation between internal and external organizational behavior and process equivalence. Although other requirements can be thought of, these are certainly among the most important. In the all above presented phenomena, the *IO π -calculus* technology can meet their desired functionalities.

3 Inter-organizational Process Advantages with π -Calculus

Since this research work is focusing on the adaptations of inter-organizational process systems to dynamic inter-organizational processes, the selected formalism must be competent by describing and coping with the dynamic evolutions of inter-organizational process systems and their dynamic processes. Coincidentally, as mentioned in requirements, the π -calculus is very suitable to describe inter-organizational process systems with changing configurations and also powerful to express the evolutions of inter-organizational process systems. So the *IO π -calculus* technology is a very natural selection used for formalizing dynamic inter-organizational processes. Due to dynamics communications in between the organizations in different situations, different organization may be used to connect. The π -calculus provides explicit transformation link during connectivity process. It can also be used to implement the transformation of import/export ports of organizations for building better connections. As far as, traditional formal methods are concerned, *IO π -calculus* technology cannot only describe actions but also reason about actions. Thus it provides us a uniform approach to specify dynamic inter-organizational processes and reason about their evolutions.

4 π -Calculus

This section presents the (context-free) grammar of the calculus, followed by the syntactic restrictions.

4.1 Syntax

The π -calculus is built around two classes of identifiers which are *Channels* and *Levels*. *Channel names* are drawn from infinite supply. Syntactically; they behave exactly as in the π -calculus *Levels* are rather more constrained; we assume prior choice of some finite and totally ordered set *level*.

Definition: (Names, processes and networks). The following grammars define the languages of processes in between networks (a machine, network or Internet).

Channels:

Host Channel $a, b ::= |() @host$

Net Channel $x, y ::= |() @host | @Net$

Names:

Meta-Variable Levels $j, k, app, host, net, \dots \in Level$

host level

S_{req} (Service Request) and $S_{getting}$ (Service getting);

net level

$w, c, r, s;$

ether (e) names:

n, p, q

Following processes are based on the asynchronous polyadic π -calculus

- 0 is inaction; it is a process that can do nothing.
- in the composition $P | Q$, the two components can proceed independently and interact via shared names or processes

- $va : \sigma.P$ denotes a process that creates fresh channel a of type σ then behave like P if v is true and behaves like Q if v is false; otherwise it is blocked forever.
- restriction $va : \sigma$; the type σ gives information about the level of operation of a and the tuples it carries.
- $!a(\vec{b}).P$ replicated input b on channel a
- $j[P]$ represents a process P running in organization at j level
- b/a replacing all free occurrences of b by a
- $fn(P)$ for the set of free names of P
- Γ is finite map from channel type names
- A type judgment is of the form $\Gamma \vdash P$ means that P is well-typed under Γ
- $[[\]]$ linkage semantic interpretations in models
- binding prefixes $a(\vec{b})$ and $!a(\vec{b})$ as input

Definition: Structural Congruence- The structural congruence relation, \equiv , is the least congruence on systems which equates Alpha-convertible systems, makes $|$ associative with identities 0 , and satisfies the following equations.

$$\begin{array}{lll}
P \mid 0 \equiv P & a(\vec{b}).P \equiv a(\vec{c}).P\{\vec{c}/\vec{b}\} & \vec{c} \cap fn(P) = \emptyset \\
P \mid Q \equiv Q \mid P & !a(\vec{b}).P \equiv !a(\vec{c}).P\{\vec{c}/\vec{b}\} & \vec{c} \cap fn(P) = \emptyset \\
(P \mid Q) \mid R \equiv P \mid (Q \mid R) & va : \sigma.P \equiv vb : \sigma.P\{b/a\} & b \notin fn(P) \\
va : \sigma.0 \equiv 0 & va : \sigma.vb : \tau.P \equiv vb : \tau.va : \sigma.P & a \neq b \\
j[va : \sigma.P] \equiv va : \sigma.(j[P]) & (va : \sigma.P) \mid Q \equiv va : \sigma.(P \mid Q) & a \notin fn(Q)
\end{array}$$

4.2 Scope and Area

The interesting equation in this structural congruence is $j[va : \sigma.P] \equiv va : \sigma.(j[P])$ commuting *name* binding and area boundaries. A consequence of this is that the scope of a *channel name*, determined by v -binding, is quite independent from the layout of areas, given by $j[-]$. Scope determines where a *name* is known, and this will change as a process evolves: areas determine how a *name* can be used, and these have a fixed structure.

For a process description to be meaningful, this fixed structure of nested areas must accord with the predetermined ordering of *levels*. For example, a *net* may contain a *host*, but not vice versa; similarly a *host* cannot contain another *host*. Writing $<_1$ for the one-step relation in the total order of *levels*, we require that in a well-formed process every nested area must be $<_1$ -below the one above. Consider some occurrence actions which are $\bar{a}(-)$, $a(-)$, or $!a(-)$ of a bound *channel name* a in a well-formed process P . The scope of a is the enclosing v -binding $va : \sigma.(-)$. The organization of this occurrence of a is the enclosing level j area $j[-]$. A single *name* may have several disjoint organizations within its scope. It is also possible for a *name* to occur outside any organization of the right level.

4.3 Type System

Channel types have the following rather simple grammar.

$$\text{Type } \sigma ::= \bar{\sigma} @ j$$

A channel type declaration of the form $a := \bar{\sigma} @ j$ states that a is a level j channel carrying tuples of values whose types are given by the vector $\bar{\sigma}$. The base types are those with empty tuples; a channel of type $() @ j$ is for synchronization within a j -area. Additional base datatypes (i.e integer or string) can be incorporated without difficulty.

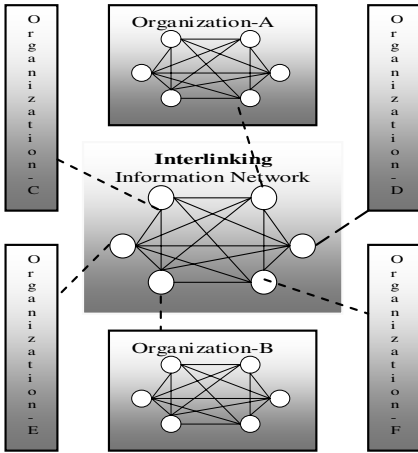


Fig. 1. Inter-organizational Process Connectivity

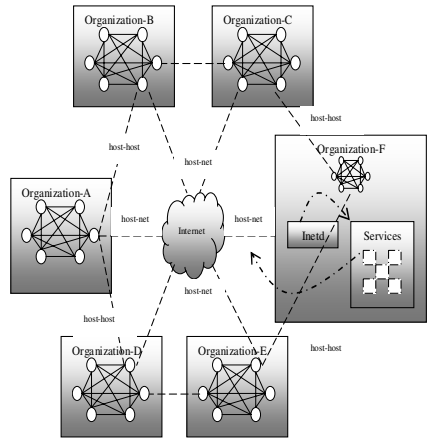


Fig. 2. Client-Server host-host and host-net level connectivity

5 IO π -Calculus Technology

In this paper, just a subset of the π -calculus to describe actions and reason about dynamic inter-organizational model is used. And thus the reasoning based on bisimilarity will turn to be simpler and possible to be automated [26]. In principle, the properties of organizations ether can be specified in terms of interface, semantics, and interaction protocol. To reason about the semantic replaceability of two pointed ethers should be the same interfaces and the consistent behavior specifications, i.e., the pre/post-conditions of the two ethers should be able to imply one another mutually. Since the semantics of the π -calculus process is donated by the evolutions (especially communications) of processes [26], the semantic replaceability of two components (Client and Server) can be reasoned about on two aspects, i.e. (1) the interfaces, i.e., the way to connect to organizations ethers, and (2) the behavior specifications, i.e., the sequences of actions exposed by the components. The presented IO π -calculus technology captures this phenomenon of names which are known universally but

always refer to local information. It extends the π -calculus so that a *channel name* can have within its scope several disjoint organizations, such a *channel name* may be used for communication within organization, or different organizations $IO\pi$ -calculus technology are arranged in a hierarchy of *levels* (between a machine, network or Internet).

The main challenge for representing $IO\pi$ -calculus technology in the plain π -calculus is how to prevent communication on a *channel* between different organizations, while still preserving the identity of *names*. Our solution is to replace communication inside or outside organization with communication on new *channel* created just for that area. As well as sending the original data, an encoded output sends the *channel name* as well.

This translation makes explicit the different *names*, identity and communication, by mapping them to two distinct sets of *names*. *Names* a and b serve purely as data, for identification. The scope of data *names* manages who knows what, while the scope of ether *names* handles locality of communication.

The output action $\bar{a}(b)$ becomes $\bar{e}(a,b)$ is a *name* that corresponds to the appropriate organization which is shared ether. Ether *names* are for communication only. An input action on a *channel* translates to a process those interactions for communication on the relevant ether. When it receives a message it tests the first element against the desired *channel name*; if they match, it accepts the input, and otherwise it rebroadcasts the message.

The evident motivation for this model is packet communication on an Ethernet [20]; instead of sending data directly to its destination, we drop a packet into the ether. Interact processes pick up all packets and shift out the ones they are interested in. this is reason, When client send request, first select appropriate servers and then establish connections for the selected servers. And when the selected servers finish providing services, it may disconnect the servers for reducing cost. In our case the tree of nested $IO\pi$ -calculus technology (machines, networks, Internet) gives rise to a hierarchy of different ethers, with a process using a different ether for each *level* of communication (a machine; Network or Internet). To improve the performance and decrease the cost of serving $IO\pi$ -calculus has the obligation to providing connections with different qualities of services for different users and servers instead of providing the same connections for all users and servers.

There is a close matching between the behavior of a process in $IO\pi$ -calculus technology and its plain π -calculus translation; here shown it is a form of weak bisimilarity on outputs. There is some loss of information though, in that translated terms may make additional silent moves, as packets pass over the ether, and ether may indiscriminately accept and rebroadcast packets in which no receiver is interested.

6 π -Calculus Within Organizations

π -calculus is a general model of computation which takes interaction as primitive. The $IO\pi$ -calculus technology extends a standard π -calculus with nested organizations locality arranged in *levels*. The π -calculus part is unexceptional: it happens to be polyadic (*channels* carry tuple values rather than single values [11]) and asynchro-

nous (output actions always succeed [22]). To illustrate the extensions, we present a brief example, based on a mechanism for selecting services. When a browser contacts a web server to fetch a page, or a person operates S (Service) to list the clients on another machine, both connect by a numbered port on the remote host for getting service. Of course, this only works if both sides agree; and there is a real-world committee to set this up [23]. There is also a further *level* of indirection; most machines run only a general meta-server *Inetd*, the Internet daemon, which listens on all ports. When *Inetd* receives a connection, it looks up the port in */etc/services*, and then consults a second file which identifies the program to provide that S service. The *Inetd* starts the program and hands it a connection to the client. There are two components involved in the *IO* π -calculus technology, i.e., the client and the server. For the client, it requests for services to the server, the server provides services.

Client $Usr = host[vc.(\bar{w}(S_{req},c) \mid c(x).\overline{S_{getting}}\langle x \rangle)]$ **Server** $Ser = host[!w(s,r).\bar{s}(r)!\overline{S_{req}}(y).\bar{y}(SerUs)]$

System $net[Usr \mid Ser]$

This shows a *client machine* $Usr(Client)$ that wishes to contact a $Ser(Server)$ with S_{req} ; the *host* $[-]$ and *net* $[-]$ indicate organizationally and inter-organizationally areas respectively. The *Client* has two components: the first transmits the S_{req} request; the second prepares to $S_{getting}$ service. *Server* Ser comprises two replicating processes: a general *Internet daemon* and a S *daemon*. *Channel* w is the internet address of the *server machine*, while the free names S_{req} and $S_{getting}$ represent well-known ports numbers. In operation, Usr sends its request to Ser naming the S_{req} service and a reply *channel* c . The *Internet daemon* on Ser handles this by retransmitting the contact c over the *channel name* S_{req} . The S *daemon* collects this and passes information on $SerUs(server User)$ back to the waiting process at Usr . A graphical interaction representation of host-host and host-net is shown as in Fig. 2.

In the plain π -calculus this models leaks: because the name S_{req} is visible everywhere, even when the *Internet daemon* on Ser has collected the request there is no protection against a S *daemon* on some different server actually handling it. Restricting the scope of S service to host Ser would be no solution, because then Usr could not formulate the request because it has to know the *name* of the service. In the *IO* π -calculus technology, each *channel* has an assigned *level* of operation, which limits how far communication on that *channel* may travel. In this case, although S_{req} is globally known, messages over it remain within a single host. Usr and Ser agree on the *name* for the S service, but different S *daemon* on separate machines does not interfere with each other.

Section 6.1 presents the rules for deriving type assertions of the form $\Gamma \vdash_j P$, this states that process P is well-typed at level j in context Γ . The static checking provided by the Type system makes two assurances one is based on the tuples values sent over *channels* will always be the right size, and the other based on a well-typed process which will not attempt to communicate on a *name* above its level of operation.

6.1 Types for Processes in the Organizational π -Calculus Technology

$$\begin{array}{c}
 \Gamma \vdash_j 0 \\
 \\
 \frac{\Gamma \vdash_j P \quad \Gamma \vdash_j Q}{\Gamma \vdash_j P|Q} \quad \frac{\Gamma, \bar{b}:\bar{\sigma} \vdash_j P}{\Gamma \vdash_j a(\bar{b}).P} \quad \Gamma(a) = \bar{\sigma} @ k \\
 \text{with } j \leq k \\
 \\
 \frac{\Gamma \vdash_j P}{\Gamma \vdash_k j|P_j} \quad j <_i k \quad \frac{\Gamma, a:\sigma \vdash_j P}{\Gamma \vdash_j va:\sigma.P} \quad \frac{\Gamma, \bar{b}:\bar{\sigma} \vdash_j P}{\Gamma \vdash_j !a(\bar{b}).P} \quad \Gamma(a) = \bar{\sigma} @ k \\
 \text{with } j \leq k \\
 \\
 \Gamma \vdash_j \bar{a}(\bar{b}) \text{ if } \Gamma(a) = \bar{\sigma} @ k, \Gamma(\bar{b}) = \bar{\sigma} \text{ and } j \leq k
 \end{array}$$

6.2 Operational Linkage for Organizational $IO\pi$ -Calculus Technology

$$\begin{array}{c}
 \text{OUT} \quad \Gamma \vdash_j \bar{a}(\bar{b}) \xrightarrow{\bar{a}(\bar{b})} 0 \\
 \\
 \text{IN} \quad \Gamma \vdash_j a(\bar{b}).P \xrightarrow{a(\bar{b})} P \quad \bar{b} \cap \text{dom}(\Gamma) = \emptyset \\
 \\
 \text{!IN} \quad \Gamma \vdash_j !a(\bar{b}).P \xrightarrow{a(\bar{b})} P|!a(\bar{b}).P \quad \bar{b} \cap \text{dom}(\Gamma) = \emptyset \\
 \\
 \text{PAR} \quad \frac{\Gamma \vdash_j P \xrightarrow{\alpha} P'}{\Gamma \vdash_j P|Q \xrightarrow{\alpha} P'|Q} \\
 \\
 \text{COMM} \quad \frac{\Gamma \vdash_j P \xrightarrow{\bar{a}(\bar{c})} P' \quad \Gamma \vdash_j Q \xrightarrow{a(\bar{b})} Q'}{\Gamma \vdash_j P|Q \xrightarrow{\varepsilon} P'|Q', \{\bar{c} / \bar{b}\}} \\
 \\
 \text{BIND} \quad \frac{\Gamma, a:\sigma \vdash_j P \xrightarrow{\alpha} P'}{\Gamma \vdash_j va:\sigma.P \xrightarrow{\alpha} va:\sigma.P'} \quad a \notin \text{fn}(\alpha) \\
 \\
 \text{AREA} \quad \frac{\Gamma \vdash_j.P \xrightarrow{\alpha} P'}{\Gamma \vdash_k j|P_j \xrightarrow{\alpha} j|P'_j} \quad \begin{array}{l} \text{if } \alpha \text{ is } \bar{a}(\bar{b}) \text{ or } a(\bar{b}) \\ \text{then } \Gamma(a) = \bar{\sigma} @ k' \\ \text{with } j <_i k \leq k' \end{array}
 \end{array}$$

6.3 Operational Links

We give the calculus a late-binding, small-step transition links, following the regular π -calculus. There is only one addition; although the static type system guarantees that a process will not initiate communication on a *name* above its operating level, we still need a dynamic check to make sure that no active communication escapes from its organization.

The operational links is given as an inductively defined relation on well-typed processes, indexed by their level j and context Γ . Section 6.2 gives rules for deriving transitions of the form $\Gamma \vdash_j P \xrightarrow{\alpha} Q$ where $\Gamma \vdash_j P$ and α (alpha) is one of the following transitions:

- $a(\bar{b}).P$ input b on channel a
- $\bar{a}(b).P$ output b on channel a
- $\tau.P$ is an invisible action to P . Can be a silent internal action of a process .

Observations. We focus on adaptations of inter-organizational system's few observations of these rules and side-conditions, such as active use of the structural congruence ' \equiv ' is essential to make full use of the rules: a process term may need to be rearranged or alpha-converted before it can make progress. For example, there is no symmetric form for the PAR rule. Secondly, in order to apply the COMM rule it may be necessary to use structural congruence to expand the scope of communicated *names* to cover both sender and recipient. Thirdly, late binding is enforced by the side-condition $\bar{b} \cap \text{dom}(\Gamma) = \emptyset$ on the input rules; this ensures that input *names* are chosen fresh, ready for substitution $Q\{\bar{c}/\bar{b}\}$ in the COMM rule. Again, we can always alpha-convert our processes to achieve this. Fourthly, the side-condition $k \leq k'$ on area is the dynamic check that prevents communications escaping from their organization.

6.4 π -Calculus Process Action Through Internet Server Daemon

$$\begin{aligned}
 \text{usr} &= \text{host}[vc : \text{response}.\overline{w}(\mathbf{S}_{\text{req}},c) \mid c(x).\overline{\mathbf{S}_{\text{getting}}} \langle x \rangle)] \\
 \text{Ser} &= \text{host}[\text{Inet} \mid \mathbf{S}_{\text{req}}] & \text{Inet} &= !w(s,r).\bar{s}\langle r \rangle \\
 \mathbf{S}_{\text{req}} &= !\mathbf{S}_{\text{req}}(y).\bar{y}\langle \text{SerUs} \rangle \\
 \Gamma &= \{ \mathbf{S}_{\text{req}} : \text{Service} & \text{service} &= \text{response} @ \text{host} \\
 & \quad w : (\text{service}, \text{response}) @ \text{net} & \text{response} &= \text{string} @ \text{net} \\
 & \quad \mathbf{S}_{\text{getting}} : \text{string} @ \text{host} \} \\
 \Gamma &\vdash_{\text{net}} (\text{Usr} \mid \text{Ser})
 \end{aligned}$$

6.5 Service Activity Through Internet Server Daemon

A host *Usr* wishes to contact a *S daemon* running on host *Ser*, through a general *Inet daemon*. Section 6.4 fills out the details of this, including type definitions.

The type service for \mathbf{S}_{req} expands to $(\text{string} @ \text{net}) @ \text{host}$.

This means that these *channels* can be used only for *host-level* communication, but the values carried will themselves be *net-level names*. The host-level communication is between *Inet* and $\mathbf{S}_{\text{service}}$ utility; the *net-level* communication is the response sent out to the original enquirer, in this case over *channel c* to *machine Usr*. *Channel w* has a *net-level* type that acts as a gateway to this, reading the *name* of a service and a *channel* where that service should send its reply.

We can now apply our operational links to see this in action.

$$\begin{aligned}
 \Gamma \vdash_{\text{net}} (\text{Usr} \mid \text{Ser}) &\equiv (\text{host}[vc : \text{response}.\overline{w}(\mathbf{S}_{\text{req}},c) \mid c(x).\overline{\mathbf{S}_{\text{getting}}} \langle x \rangle] \mid \text{host}[\text{Inet} \mid \mathbf{S}_{\text{req}}]) \\
 \text{Extended scope of } c &\equiv vc : \text{response}.\overline{w}(\mathbf{S}_{\text{req}},c) \mid c(x).\overline{\mathbf{S}_{\text{getting}}} \langle x \rangle \mid \text{host}[\text{Inet} \mid \mathbf{S}_{\text{req}}]
 \end{aligned}$$

communications on:

- 1- $w @ net \xrightarrow{\tau} vc : response.(host[c(x).\overline{S}_{getting}\langle x \rangle] | host[\overline{S}_{req}\langle c \rangle] | Inet | S_{req})$
- 2- $S_{req} @ host \xrightarrow{\tau} vc : response.(host[c(x).\overline{S}_{getting}\langle x \rangle] | host[Inet | \overline{c}\langle SerUs \rangle] | S_{req})$
- 3- $c @ net \xrightarrow{\tau} vc : response.(host[\overline{S}_{getting}\langle SerUs \rangle] | host[Inet | S_{req}])$

expands of:

- 1- $Inet \equiv vc : response.(host[\overline{w}\langle S_{req}, c \rangle] | c(x).\overline{S}_{getting}\langle x \rangle] | host[!w(s, r).\overline{s}\langle r \rangle] | S_{req})$
- 2- $S_{req} \equiv vc : response.(host[c(x).\overline{S}_{getting}\langle x \rangle] | host[\overline{S}_{req}\langle c \rangle] | Inet | S_{req}(y).\overline{y}\langle SerUs \rangle])$

After a sequence of internal communications at the *net* and *host* level, the first *host Usr* is ready to get the service information *SerUs*, and *host Ser* is restored to its original configuration.

7 Asynchronous π -Calculus as a Target π -Calculus

The target calculus is an asynchronous π -calculus [22], with guarded recursion and name testing.

$$P ::= a(b).P | \overline{a}\langle b \rangle | 0 | (P | Q) | \mu X.P | vx.P | \text{if } x = y \text{ then } P \text{ else } Q$$

The absence of output prefixing and choice reflects the organization calculus. Other aspects are tuned to make the encoding as simple as possible by reducing internal transitions and avoiding inert processes. For example, we could easily use replication $!P$ rather than recursion, but this needs an extra trigger *channel*. Unusually, our *channels* carry non-empty lists of values rather than tuples; we write b for this, with; for list concatenation. This is because the translation multiplexes the action of several polyadic *organizational π -channels* onto single, and hence a single π -channel may carry packets of different sizes. We use head/tail pattern matching on these lists to unwrap packets; in fact, testing on the head element of a list is always enough to determine its length. An alternative would be to further encode lists using standard π -calculus techniques [24], or possibly type packets with some polymorphic data type [25]. We need to test *names* for both equality and inequality, and so combine these into a conditional with operational rules derived from those for matching [26].

$$\frac{P \xrightarrow{\alpha} P'}{\text{if } x = x \text{ then } P \text{ else } Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{\text{if } x = y \text{ then } P \text{ else } Q \xrightarrow{\alpha} Q'} \quad x \neq y$$

With these rules we can consistently add the following convenient structural congruence: $(\text{if } x = x \text{ then } P \text{ else } Q) \equiv P$.

8 Compositional Encoding of Organizational π -Term into the π -Calculus

In this section we represent a compositional encoding of organization π -term into the π -calculus, following the scheme outlined in the introduction. All communication is mapped into packets passing over designated ether *channels*; thus tuple output $\overline{a}\langle \overline{b} \rangle$ becomes list output $\overline{e}\langle a; b \rangle$ where e varies according to the level of a . To keep track of which e to use, we maintain an environment Δ mapping *levels* to ether *names*. The

encoding is parameterized over this, and takes the form $\llbracket \Gamma \vdash_j P \rrbracket_\Delta$ where Δ assigns ethers to *levels* j and above.

Section 8.1 presents the full encoding, with one clause for each constructor; here we go through each one individually. The null process, parallel composition, and *name* restriction are unchanged.

$$\begin{aligned} \llbracket \Gamma \vdash_j 0 \rrbracket_\Delta &= 0, & \llbracket \Gamma \vdash_j P \mid Q \rrbracket_\Delta &= \llbracket \Gamma \vdash_j P \rrbracket_\Delta \mid \llbracket \Gamma \vdash_j Q \rrbracket_\Delta \text{ and} \\ \llbracket \Gamma \vdash_j \nu a : \sigma.P \rrbracket_\Delta &= \nu a. \llbracket \Gamma, a : \sigma \vdash_j P \rrbracket_\Delta \end{aligned}$$

To place a process in an organization, as an agent, we create a new ether *name* and assign it a *level* in the environment Δ . A side condition ensures that we do not accidentally capture any existing *names* when introducing the new ether.

$$\llbracket \Gamma \vdash_j k[P] \rrbracket_\Delta = \nu e. \llbracket \Gamma \vdash_k P \rrbracket_{\Delta, k \mapsto e} \quad e \notin \text{fn}(P) \cup \text{cod}(\Delta)$$

Translating service getting action uses the environment and the assignment of *levels* to ethers to find the correct ether for the output *channel*. It then sends both the output *channel* and the data for transmission over this ether *name*.

$$\llbracket \Gamma \vdash_j \bar{a}(b) \rrbracket_\Delta = \bar{e}(a; b) \text{ with } e = \Delta(k) \text{ where } \Gamma(a) = \bar{\sigma} @ k$$

An encoded input also uses the environment and the assignment of *levels* to ether *names* to find which ether it should listen on. When it receives a packet over this ether, it tests the head of the list to see if it matches the input *channel name*. If it does, then the packet is meant for this input and execution continues as appropriate. If the *names* do not match, then this packet is meant for some other *channel* in the same area. The packet is resent and the process restarts.

$$\llbracket \Gamma \vdash_j a(\bar{b}).P \rrbracket_\Delta = \mu X. e(x; b). \text{if } x = a \text{ then } \llbracket \Gamma, b : \bar{\sigma} \vdash_j P \rrbracket_\Delta \text{ else } \bar{e}(x; b) \mid X \text{ with } e = \Delta(k) \text{ where } \Gamma(a) = \bar{\sigma} @ k$$

Replicated input is the same, except that the process restarts whether or not the input key is correctly matched.

$$\llbracket \Gamma \vdash_j !a(\bar{b}).P \rrbracket_\Delta = \mu X. e(x; b).(X \mid \text{if } x = a \text{ then } \llbracket \Gamma, b : \bar{\sigma} \vdash_j P \rrbracket_\Delta \text{ else } \bar{e}(x; b)) \text{ with } e = \Delta(k) \text{ where } \Gamma(a) = \bar{\sigma} @ k.$$

The encoding is well-defined up to structural congruence.

Proposition. For any organization π -term P and Q , if $P \equiv Q$ then $\llbracket P \rrbracket_\Delta \equiv \llbracket Q \rrbracket_\Delta$

Proof: Because the encoding is compositional, it is enough to check that all of the structural axioms for $IO\pi$ -calculus technology given at the end of Section 4, translate to valid π -calculus equivalences. All of these are immediate; the only significant case is that $j[\nu a : a.P] \equiv \nu a : \sigma.(j[P])$ becomes exchange of *name* binders $\nu e. \nu a . \llbracket P \rrbracket_\Delta \equiv \nu a. \nu e. \llbracket P \rrbracket_\Delta$ for some ether e . It is worthwhile noting that the encoding uses π -calculus *channels* in a highly manner. *Names* fall into two distinct classes, data *names* which correspond directly to organization π -calculus *channels*, and ether *names*. All communication involves sending data over ethers. One consequence of this is that all our terms happen to lie in the subset studied by Massimo Merro [27] as the $IO\pi$ -calculus technology, where *names* sent over *channels* may not be used for further communication. In section 4 and 5, we mentioned that in general π -calculus *names* have a dual role, for identity and for communication; what happens in the translation is that each role is mapped to a different *name*.

8.1 Rules for Encoding Organizational π -Calculus Technology with π -Calculus

- (i) $\llbracket \Gamma \vdash_j 0 \rrbracket_\Delta = 0$
- (ii) $\llbracket \Gamma \vdash_j P \mid Q \rrbracket_\Delta = \llbracket \Gamma \vdash_j P \rrbracket_\Delta \mid \llbracket \Gamma \vdash_j Q \rrbracket_\Delta$
- (iii) $\llbracket \Gamma \vdash_j k[P] \rrbracket_\Delta = \nu e. \llbracket \Gamma \vdash_k P \rrbracket_{\Delta, m \rightarrow e} \quad e \notin \text{fn}(P) \cup \text{cod}(\Delta)$
- (iv) $\llbracket \Gamma \vdash_j \nu a : \sigma. P \rrbracket_\Delta = \nu a. \llbracket \Gamma, a : \sigma \vdash_j P \rrbracket_\Delta$
Actions, all with $e = \Delta(k)$ where $\Gamma(a) = \bar{\sigma} @ k$:
- (v) $\llbracket \Gamma \vdash_j \bar{a}(\vec{b}) \rrbracket_\Delta = \bar{e}(a; b)$
- (vi) $\llbracket \Gamma \vdash_j a(\vec{b}).P \rrbracket_\Delta = \mu X.e(x; b).(if \ x = a \ \text{then} \ \llbracket \Gamma, b : \bar{\sigma} \vdash_j P \rrbracket_\Delta$
 $else \ \bar{e}(x; b) \mid X)$
- (vii) $\llbracket \Gamma \vdash_j !a(\vec{b}).P \rrbracket_\Delta = \mu X.e(x; b).(X \mid if \ x = a \ \text{then} \ \llbracket \Gamma, b : \bar{\sigma} \vdash_j P \rrbracket_\Delta$
 $else \ \bar{e}(x; b))$

9 Encoding Correctness for Organizational π -Process

Our main result is in the form of bi-similarity on outputs, up to the translation between direct and ether-based communication. Organization π -process and its encoding behave in very similar ways, and this is preserved under reduction relation, \Rightarrow , is the least relation on systems satisfying the following equations.

Theorem. For any well-typed process $\Gamma \vdash_j P$ in the $IO\pi$ -calculus technology and transition $\alpha = \bar{a}(\vec{b})$ or $\alpha = \tau$ the following hold.

- (i) If $\Gamma \vdash_j P \xrightarrow{\alpha} P'$ then $\llbracket \Gamma \vdash_j P \rrbracket_\Delta \xrightarrow{\llbracket \alpha \rrbracket_\Delta} \llbracket \Gamma \vdash_j P' \rrbracket_\Delta$
- (ii) If $\llbracket \Gamma \vdash_j P \rrbracket_\Delta \xrightarrow{\llbracket \alpha \rrbracket_\Delta} Q$ then there is P' such that $\Gamma \vdash_j P \xrightarrow{\alpha} P'$ and $\llbracket \Gamma \vdash_j P' \rrbracket_\Delta \equiv Q$.

Here $\llbracket \tau \rrbracket_\Delta = \tau$ and $\llbracket \bar{a}(\vec{b}) \rrbracket_\Delta = \bar{e}(a, b)$ where $e = \Delta(k)$ for $\Gamma(a) = \bar{\sigma} @ k$

This result says nothing about inputs; in fact a term and its encoding generally have different input behavior, with the translated terms being more receptive than the original. It is conventional though in asynchronous calculi to regard only output as observable, as there is no way in principle to know when an input has been received. In the terminology of Nestmann and Pierce [28], this is an operational correspondence between the calculi. Although expressed in terms of weak transitions, the correspondence is in fact rather close. Transitions match exactly, except that a single internal π -transition may map to zero organization π -transitions. In proposed system, divergence arises rather naturally from mechanism of ethers. We expect that replacing this with more pragmatic lists of service user and service provider would lead to a divergence-free encoding, but at a cost of considerable complexity. *Theorem* follows with-

out difficulty from the following more precise results, which characterize exactly the possible actions of encoded processes.

Lemma. For any well-typed process $\Gamma \vdash_j P$ in the $IO\pi$ -calculus technology the following clauses hold. In each case $Q = \llbracket \Gamma \vdash_j P \rrbracket_\Delta$ and $e = \Delta(k)$ where $\Gamma(a) = \bar{\sigma} @ k$. Following clauses make clear the close connection between organization π -transitions and ether packets. For output, the correspondence of process can perform an output if and only if its translation can. For input, recall that when an encoded process reads a packet, it tests it and if unsuitable it retransmits the packet again and continues as before. This means that an input in the encoded system may be matched by a similar input in the system it encodes or it may perform an output and revert to the original process. This choice of two possible responses to any input action is carried out to the case of an encoded process performing a τ . This may either reflect a τ in the system it encodes or it may be a rejected communication, in which case the process it reduces to its congruence to the original.

- (i) If $\Gamma \vdash_j P \xrightarrow{\bar{a}(\bar{b})} P'$ then $Q \xrightarrow{\bar{e}(a,b)} Q'$ where $Q' \equiv \llbracket \Gamma \vdash_j P' \rrbracket_\Delta$
- (ii) If $\Gamma \vdash_j P \xrightarrow{a(\bar{b})} P'$ then $Q \xrightarrow{e(x,b)} Q'$ such that any vector of names \vec{c} we have $Q'\{a; c/x; b\} \equiv \llbracket \Gamma' \vdash_j P' \{\vec{c}/\bar{b}\} \rrbracket_\Delta$
- (iii) If $\Gamma \vdash_j P \xrightarrow{\tau} P'$ then $Q \xrightarrow{\tau} Q'$ where $Q' \equiv \llbracket \Gamma \vdash_j P' \rrbracket_\Delta$.
- (iv) If $\Gamma \vdash_j Q \xrightarrow{\bar{e}(a,b)} Q'$ then $\Gamma \vdash_j P \xrightarrow{\bar{a}(\bar{b})} P'$ with $\llbracket \Gamma \vdash_j P' \rrbracket_\Delta \equiv Q'$
- (v) If $Q \xrightarrow{e(x,b)} Q'$ then either $Q' \equiv \bar{e}(x; b) | Q$ or $\Gamma \vdash_j P \xrightarrow{a(\bar{b})} P'$ where for any vector of names \vec{c} we have $Q'\{a; c/x; b\} \equiv \llbracket \Gamma' \vdash_j P' \{\vec{c}/\bar{b}\} \rrbracket_\Delta$
- (vi) If $Q \xrightarrow{\tau} Q'$ then either $Q \equiv Q'$ or $\Gamma \vdash_j P \xrightarrow{\tau} P'$ with $\llbracket \Gamma \vdash_j P' \rrbracket_\Delta \equiv Q'$

Proofs for each above clauses in the lemma follow a similar pattern. For clauses (i)-to-(iii), we break down a process into the part that performs the action and a surrounding context. Next we use the encoding rules to encode these parts. Then we show how the encoding of the part of the organization π -process that performs the action can perform a matching π -action. Finally, we show that the encoding of the context allows this similar action to escape. There is a dependency, in that we must prove parts (i) and (ii) before (iii). Clauses (iv)–(vi) are proved similarly, but in the reverse direction. First we break down Q , into the parts that perform the action and a context, then using this decomposition we characterize P , finally we show that this P can perform the required action and reduce to a process matching Q' . The direct relationship between the behavior of a process and its encoding make the proof much easier. In particular, there are no intermediate forms on the π -calculus side to be analyzed. If there were such additional “housekeeping” steps, then we would need to enlarge the lemma to cover a one-to-many relation R_Δ between organization π -processes and π -term.

9.1 Encoding Correctness Processes Action Through Internet Server Daemon

$$\begin{aligned}
& \llbracket \Gamma \vdash_{net} U_{sr} \mid Ser \rrbracket \{_{net} \mapsto n \} = U_{sr}' \mid Ser' \\
& U_{sr}' = \nu q.vc.(\bar{n}\langle w, \mathbf{S}'_{req}, c \rangle \mid \mu X.n(x; y).(if\ x = c\ then\ \bar{q}\langle \mathbf{S}'_{getting}; y \rangle\ else\ \bar{n}\langle x; y \rangle \mid X)) \\
& Ser' = \nu p.(Inet' \mid \mathbf{S}'_{req}) \\
& Inet' = \mu X.n(x; y).(X \mid if\ x = w\ then\ \bar{p}\langle y \rangle\ else\ \bar{n}\langle x; y \rangle) \\
& \mathbf{S}'_{req} = \mu X.p(s, r).(X \mid if\ s = \mathbf{S}_{req}\ then\ \bar{n}\langle r, SerUs \rangle\ else\ \bar{p}\langle s, r \rangle)
\end{aligned}$$

10 Encoding of the Internet Server Daemon

To illustrate how this works we encode the *Inet daemon* example from Section 6.4. Section 9.1 shows the result, which can be compared with section 6.5. The translation uses three ethers, for which we take names n , p and q to cover the network, server host Ser and client host U_{sr} respectively. All organization π -channel names map to themselves.

As we expect from Lemma, reductions of the translated process closely match those of the original given earlier.

$$\begin{aligned}
& U_{sr}' \mid Ser' \equiv (\nu q.vc.(\bar{n}\langle w, \mathbf{S}'_{req}, c \rangle \mid \mu X.n(x; y)..... \\
& \quad \mid \nu p.(Inet' \mid \mathbf{S}'_{req}))) \\
& \text{extended scope of } p, q \text{ and } c \equiv \nu p, q, c.(\bar{n}\langle w, \mathbf{S}'_{req}, c \rangle \mid \mu X.n(x; y)..... \\
& \quad \mid Inet' \mid \mathbf{S}'_{req}) \\
& \text{Unroll } Inet' \equiv \nu p, q, c.(\bar{n}\langle w, \mathbf{S}'_{req}, c \rangle \mid \mu X.n(x; y)..... \\
& \quad \mid n(x; y).(Inet' \mid if\ x = w\ then\ \bar{p}\langle y \rangle\ else\ \bar{n}\langle x; y \rangle) \mid \mathbf{S}'_{req}) \\
& \text{Communication of } w \text{ over } n \xrightarrow{\tau} \nu p, q, c.(\mu X.n(x; y)..... \\
& \quad \mid if\ w = w\ then\ \bar{p}\langle \mathbf{S}'_{req}, c \rangle\ else..... \\
& \quad \mid Inet' \mid \mathbf{S}'_{req}) \\
& \text{Apply test and Unroll } \mathbf{S}'_{req} \equiv \nu p, q, c.(\mu X.n(x; y)..... \\
& \quad \mid \bar{p}\langle \mathbf{S}'_{req}, c \rangle \mid p(s, r).(\mathbf{S}'_{req} \mid p(s, r).(\mathbf{S}'_{req} \mid if\ s = \mathbf{S}_{req}\ then\ \bar{n}\langle r, SerUs \rangle\ else.....) \\
& \quad \mid Inet')) \\
& \text{Communication of } \mathbf{S}'_{req} \text{ over } p \xrightarrow{\tau} \nu p, q, c.(\mu X.n(x; y)..... \\
& \quad \mid if\ \mathbf{S}'_{req} = \mathbf{S}_{req}\ then\ \bar{n}\langle c, SerUs \rangle\ else..... \\
& \quad \mid Inet' \mid \mathbf{S}'_{req}) \\
& \text{Apply test} \equiv \nu p, q, c.(\mu X.n(x; y).if\ x = c\ then\ \bar{q}\langle \mathbf{S}'_{getting}; y \rangle\ else..... \\
& \quad \mid \bar{n}\langle c, SerUs \rangle \mid Inet' \mid \mathbf{S}'_{req}) \\
& \text{Communication of } c \text{ over } n \xrightarrow{\tau} \nu p, q, c.(if\ c = c\ then\ \bar{q}\langle \mathbf{S}'_{getting}, SerUs \rangle\ else... \\
& \quad \mid Inet' \mid \mathbf{S}'_{req}) \\
& \text{Apply test} \equiv \nu p, q, c.(\bar{q}\langle \mathbf{S}'_{getting}, SerUs \rangle \mid Inet' \mid \mathbf{S}'_{req})
\end{aligned}$$

Comparing the reduction in given in Section 6.4, notice how communication restricted to an organization (“communication on $S_{req} @ host$ ”) is replaced by communication on organization ether (“communication of S_{req} over p ”).

11 Conclusion

A new formalizing approach presented for modeling and controlling the execution of inter-organizational service processes by *IO π -calculus* technology. The inter-organizational process used as host-host and host-net to refer to a specified process service. Each one points to or provided by the intra-process of an organization and can be changed dynamically by adapting business process environment. We also defined the syntax, the operational linkage, and a type system for an asynchronous and distributed π -calculus with local and global communication and process migration. We have encoded a notion of distributed organizational process system and inter-organizational communication into the π -calculus by giving a translation of the *IO π -calculus* technology. *IO π -calculus* technology encoding is the technique of replacing communication on a *channel name* with communication over ether associated with the appropriate organizational process system. In the operational correspondence section 9, we showed that there is a close relation between the actions of processes and their translations. This step used to build on to investigate the degree to which the encoding preserves and reflects equivalences in between inter-organizational processes. We would expect adequacy, but not full abstraction, as encoding organization by ethers exposes them to probing by general π -calculus terms. The encoding given in the present paper assumes that each process has direct access to the ethers for every containing *level*. We can suggest a solution though, using an encoding with network controllers. Within organization, each process communicates only through its immediate organizational controller. This approach is well suited for sporadic communication in open organizational distributed systems, especially co-operations across organization and inter-organizational boundaries. The cooperative process across organizations becomes simple, faster, and flexible. The new internet cooperative system presents the construction of an internet/inter-organizational process. Hence, applying excepting handling and transaction mechanisms of processes into inter-organizational process system are future topics.

References

1. Rainer Schmidt, “ Web Services Based Architectures to Support Dynamic Inter-organizational Business Processes”. ICWS-Europe 2003, LNCS 2853, pp. 123–136. © Springer-Verlag Berlin Heidelberg 2003.
2. Bettina Bazijanec, Christian Winnewisser, Antonia Albani, and Klaus Turowski, “A Component-Based Architecture for Protocol Vector Conversion in Inter-organizational Systems”, OTM Workshops 2004, LNCS 3292, pp. 556–567, 2004. © Springer-Verlag Berlin Heidelberg 2004.
3. Philippa Gardner, Sergio Maffei, “Modelling Dynamic Web Data?”, Preprint submitted to Elsevier Science 21 December 2004.

4. John E. Taylor, Raymond E. Levitt, "Inter-organizational Knowledge Flow and Innovation Diffusion in Project-based Industries". Proceedings of the 38th Hawaii International Conference on System Sciences. © 2005 IEEE.
5. K. Dittrich and D. Tombros, "Workflow Management for the Virtual Enterprise". *Proc. of the International Process Technology Workshop*, 1999.
6. Olivier Perrin, Claude Godart, "A model to support collaborative work in virtual enterprises". © Elsevier -2004.
7. Wil van der Aalst, "Loosely coupled interorganizational workflows: modeling and analyzing workflows crossing organizational boundaries". © Elsevier Science 2000.
8. Gwen. Salaiün, L. Bordeaux, Marco schaeerf, "Describing and Reasoning on Web Services using Process Algebra". *Proc. of the International Conference on Web Services* © IEEE 2004.
9. Liu Shuzhou and Angela Goh Eck Soong, "A Formal Framework to Support Workflow Adaptation". *International Journal of Software Engineering and Knowledge Engineering*, 2002, 12(3): 245-2687.
10. Rachid Hamadi and Boualem Benatallah, "A Petri net-based model for web service composition", *Proc. of the 14th Australasian database conference on Database technologies*, © Australian Computer Society, Inc. 2003.
11. Robin Milner, "The polyadic π -calculus: a tutorial. Laboratory for Foundations of Computer Science , Computer Science Department, University of Edinburgh, The King's Buildings, Edinburgh, UK, October 1991.
12. Benjamin Pierce and Davide Sangiorgi, "Typing and subtyping for mobile processes". In *Logic in Computer Science*, pages 187-215, 1993.
13. Benjamin Pierce and D. Turner, "Pict: a programming language based on the π -calculus", Indiana University, CSCI Technical Report March 19,1997 .
14. José-Luis Vivas and Mads Dam, "From higher-order π -calculus to π -calculus in the presence of static operators". In *CONCUR '98: Concurrency Theory. Proceedings of the 9th International Conference*, Lecture Notes in Computer Science 1466. © Springer-Verlag, 1998.
15. Luca Cardelli, Giorgio Ghelli, and Andrew D. Gordon, "Secrecy and group creation". In *CONCUR '2000: Concurrency Theory. Proceedings of the 11th International Conference*, Lecture Notes in Computer Science 1877, pages 365–379. © Springer-Verlag, 2000.
16. C. Fournet, G. Gonthier, J.J. Lévy, L. Maranget, D. Rémy, "A calculus of mobile agents", in: C.A. Gunter, J.C. Mitchell (Eds.), *CONCUR'96*, Pisa, Italy, Springer, Berlin, 1996.
17. Luca Cardelli, A.D. Gordon, "Mobile ambients", in: M. Nivat (Ed.), *FoSSaCS*, Lecture Notes in Computer Science, Vol. 1378, Springer, Berlin, Germany, 1998, pp. 140–155.
18. R. M. Amadio, "On modelling mobility", Preprint submitted to Elsevier, 15 Feb. 1999.
19. Matthew Hennessy, J. Riely, "Resource access control in systems of mobile agents", the 3rd International Workshop on High-Level Concurrent Languages (HLCL'98), vol.16(3) of *Electronic Notes in Theoretical Computer Science*. © Elsevier, 1998
20. Jinggang Wang and Binoy Ravindran, "Packet Scheduling Algorithm, Implementation, and Feasibility Analysis", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, VOL. 15, NO. 2, FEBRUARY 2004 119. © 2004 IEEE.
21. Wil van der Aalst "Process-oriented Architectures for Electronic Commerce and Interorganizational Workflow. *Information Systems*", 1999, 24(8): 639-671.
22. Gérard Boudol, "Asynchrony and the π -calculus (Note)". Rapport de recherche 1702, INRIA, Sophia Antipolis, 1992.
23. IANA, "the Internet Assigned Numbers Authority. Protocol numbers and assignment services: Port numbers". <http://www.iana.org/numbers.html>.

24. David N. Turner, "*The Polymorphic π -calculus: Theory and Implementation*". PhD thesis, Laboratory for Foundations of Computer Science, Edinburgh University, 1996. Also published as LFCS Technical Report 345.
25. Benjamin Pierce and Davide Sangiorgi, "Behavioral equivalence in the polymorphic π -calculus". *Journal of the ACM*, 47(5):531–584, September 2000.
26. Robin Milner, Joachim Parrow, and David Walker, "A calculus of mobile processes, parts I and II". *Information and Computation*, 100:1–77, 1992.
27. Massimo Merro, "*Locality in the π -calculus and applications to distributed objects*". PhD thesis, Ecole des Mines, France, 16 October 2000.
28. Uwe Nestmann and Benjamin C. Pierce, "Decoding choice encodings". BRICS report RS-99-42, Department of Computer Science, University of Aarhus, December 1999. To appear in *Information and Computation*; a version appeared as a paper at CONCUR '96.

Modulation for Scalable Multimedia Content Delivery*

Henry Novianus Palit¹ and Chi-Hung Chi²

¹ School of Computing, National University of Singapore, Singapore 117543

² School of Software, Tsinghua University, Beijing, China 100084
chichihung@mail.tsinghua.edu.cn

Abstract. In this paper, we propose a new approach to perform real-time transcoding of images for pervasive Internet access. Taking full advantages of the trend of new image format and structure, this new transcoding mechanism, which we call it modulation, achieve the followings: (a) negligible overhead in real-time transcoding, (b) on-demand bandwidth consumption between the web server and the client/proxy, (c) maximum reuse of cached image variations, independent of whether it is from lower quality to higher quality or vice versa. Our result shows that this modulation approach gives full potentials for real-time image transcoding for pervasive Internet access.

1 Introduction

The increasing heterogeneity of client's device, browsing preference, and network connectivity results in a big challenge of pervasive Internet access. The traditional approach is to offer multi-version web pages, each version for a specific type of client devices. This approach is inflexible and inefficient. Changing the web pages is quite troublesome since any change on a particular version must be replicated to the other versions. It is also costly, as more disk space is required to store the multiple versions. In order to reduce the disk space usage, often the number of versions is restricted. Hence, the given web service becomes inflexible.

Transcoding (sometimes called *distillation*), the recent proposal to address client heterogeneity, is defined as the process to convert a representation of a data object into another representation [6]. A typical transcoding system works as follows: the transcoding system – usually embedded in a web intermediary – retrieves a web object on client's behalf, transcodes the object accordingly, and then sends the transcoded object to the client. Although transcoding offers flexibility and may not need much disk space, it still exhibits many weaknesses, some of the most important ones are server to proxy bandwidth usage, transcoding overhead, and data reusability among transcoded versions of the same object.

The introduction of a new image standard, JPEG2000 [7], inspired us to propose a new technique to serve pervasive web access more efficiently. Besides superior low bit-rate performance and robustness to bit-errors, JPEG2000 supports progressive transmission [3], in which an image is transferred packet by packet resulting in a gradual display of the image. The gradual display can be performed in pixel accuracy

* This research is supported by the funding 2004CB719400 of China.

(blurred to clear presentation) and/or in resolution (small to large presentation). In the JPEG2000 standard, an image consists of several layers of quality and several levels of decomposition. Based on the quality layers, the decomposition levels, and two other parameters (color components and precincts), the image is partitioned into packets. Dropping packets representing high quality layers from a JPEG2000 image will give us a low quality (blurred) image. Similarly, dropping packets representing high decomposition levels will result in a low resolution (small) image. The quality and resolution reduction can be performed gradually depending on the number of quality layers and decomposition levels, respectively. Of one image, different representations can be generated without complex computations. Moreover, a low quality or low resolution representation can be improved by adding the necessary packets. If all these features are applied to the Web, we will get a new transcoding approach with the following characteristics: able to work with partial objects (i.e., packets), no complex computations, and high data reuse.

In this paper, we propose a new approach to perform real-time transcoding of images for pervasive Internet access. Taking full advantages of the trend of new image format and structure, this new transcoding mechanism, which we call it modulation, achieve the followings: (a) negligible overhead in real-time transcoding, (b) on-demand bandwidth consumption between the web server and the client/proxy, (c) maximum reuse of cached image variations, independent of whether it is from lower quality to higher quality or vice versa.

2 Related Work

Many transcoding systems have been built to meet the demand for pervasive Internet access. The representative ones are proxy-based GloMop [4] and Mowser, and server-based system InfoPyramid [11]. Most of them use implicit information, such as the HTTP Content-Type header, to control when and how they convert between representations. Mogul et al. [8] [10] argued that such approach could risk the loss of important information since the proxy did not know the semantics of the content. They proposed a server-directed transcoding (SDT) to preserve end-to-end semantics while supporting aggressive content transformations. Han et al. [6] presented an analytical framework for determining whether to transcode and how much to transcode an image for the cases of store-and-forward transcoding as well as streamed transcoding. Chandra and Ellis [1] a quality aware image transcoding based on JPEG Quality Factor. Integration of transcoding and caching systems was done by TranSquid [9]. TranSquid maintains separate caches for different categories of clients (high capability, medium capability, and limited capability).

Deshpande and Zeng [5], Taubman [12], and our previous work [2] made use of JPEG2000 to provide scalable image presentation. Various representations can be generated only by virtue of requesting necessary parts of a JPEG2000 image. This shows the advanced features of the JPEG2000 standard, compared to the other image standards. While the other two studies focused on fulfilling a client's needs, our previous work tried to cover a wider client base with the help of the caching system. Our work can achieve high data reuse while keeping only a single variant in the caching system. However, due to its ignorance about the JPEG2000 data format, it

suffers some drawbacks. Sometimes the image presentation is split into two regions, one with low quality and the other with high quality. In addition, our previous work can only exploit the image scalability in quality, whereas JPEG2000 also support scalability in resolution.

3 Modulation

The Oxford English Dictionary¹ defines modulation as the action of forming, regulating, or varying according to due measure and proportion. In this paper, we define *modulation* as the process to obtain an object's representation by means of adjusting (eliminating and/or adding) the building blocks of the object. The building blocks of an object could be layers, packets, fragments, and so forth (for simplicity, we use fragments to represent the object's building blocks for the rest of this section). A typical modulation system works as follows:

1. Receiving a client request, a modulating proxy adapts the request to fit the client constraints. Alternatively, the client may have specified its preferences in the request. The proxy then looks for the requested object in its cache.
 - a. If the outcome is a *hit*, the proxy examines whether the cached variant can fulfill the client request. If it can, continue to step 3. If it can only partly fulfill the request (called a *partial hit*) and therefore, some object fragments are still required, a request for those fragments is sent to the web server.
 - b. Otherwise (it is a *miss*), the adapted request is passed on to the web server.
2. The web server modulates the requested object according to the (adapted) client request. The modulated output, which does not include fragments of the object deemed unneeded, is sent back to the proxy, most likely in a datastream.
3. If an object variant, which can fulfill the client request either entirely or partly, exists in its cache, the proxy can modulate the variant, if necessary, and stream the variant (or the modulated one) to the client while waiting for the other fragments of the object coming from the web server. Hence, minimal client latency can be sustained. Besides being streamed to the client, the object fragments received from the web server are either stored in the cache individually or combined with the cached variant, if any. For every web object, only a single variant should be kept in the cache all the time, and there is no duplication of object fragment in the cache.

3.1 Characteristics

Modulation is expected to deliver proper object's representations to heterogeneous clients fast and efficiently. Based on the general steps described above, modulation has the following characteristics:

- Modulation is basically collaboration between web servers and caching proxies. The caching system is attached to the (forward) proxies, which are close to the clients, so that the client requests can be served locally. This will reduce the

¹ The Oxford English Dictionary, 2nd edition; prepared by J. A. Simpson and E. S. C. Weiner; Vol. IX, page 955; Oxford University Press, 1989.

bandwidth consumption as well as the client latency. The modulating system is attached to both the web servers and the proxies. Object modulation should begin as early as possible from the original web servers in order to efficiently deliver on-demand object fragments. Hence, more reduction in the bandwidth consumption and the client latency can be expected.

- Modulation is an exclusive process. It is an object transformation within a data format. If a client browser does not support the modulated data format, then transcoding is still required for conversion between data formats. This may be the only drawback of modulation.
- Unlike transcoding, which only transforms a high fidelity variant to a low fidelity variant, modulation is a reversible² process. An object's representation can be generated by eliminating a few fragments of the original object. Conversely, the original object can still be retrieved by adding the missing fragments to the representation. The reversible property makes high data reuse possible in modulation.
- Since modulation involves only elimination and addition of an object's fragments, and no complex computation whatsoever, the process is fast and the workload minimal. That is why it can be performed at the web server without much decrease in performance.

Preserving the end-to-end semantics should not be an issue in modulation. During the creation of the object, the degree of details in each level of presentation can be specified. The essential semantics of the object should be retained in all presentation levels. So, when the object is modulated, the modulating system will not care about the object semantics anymore. If necessary, the object may be accompanied with additional hints to modulate the object appropriately. SDT [10] can be a good choice to deliver the hints.

3.2 Conceptual Framework

We present the conceptual framework of modulation to bring clarity of the transformations within it. The framework is centered on the data model since it is the most important element of modulation. The efficiency of data delivery achieved in modulation is mainly due to the data model. On the other hand, modulating processes are relatively simple, albeit essential, too. The last element of modulation is metadata, whose information can help to modulate the associated object properly. The data model, the modulating processes, and the metadata are elaborated in the following subsections.

3.2.1 Data Model

As defined earlier, modulation is an eliminating and/or adding process applied to the building blocks (fragments) of an object, in order to obtain the appropriate object's representation. The fragments can be differentiated one another because each of them has distinct attribute values. The attributes (or properties) can be anything, depending

² Another definition of modulation in The Oxford English Dictionary:
Biol. Reversible variation in the activity or form of a cell in response to a changing environment.

on the object’s data format. An image, for instance, may specify color component and quality degree as the attributes of its fragments. Color images in almost all known image formats have at least 3 (three) color components, and the data supporting each color component are usually easy to spot. So, we can gather data fragments of an image based on the color component. In a multi-scan image format, like progressive JPEG and interlaced GIF, each scan adds to the image clarity (or quality). The data associated with each scan can also be identified, and therefore, retrieving data fragments based on the quality degree is possible, too. If some data fragments, either based on color component or quality degree, can be dropped from an image without the need for recoding and the resulted image is still presentable – although it is of low quality –, that would be ideal for modulation.

The fragments of an object in the data model are *segments* and *atoms*. An object is composed of segments. As pointed out before, the division of an object into segments is based on one or more specified attributes. Thus, each segment carries certain attribute values, which can distinctly identify the segment. Each segment is further composed of atoms. An atom is the smallest entity, which is indivisible. More attributes are required to identify an atom uniquely. We define two types of fragments – segment and atom – in the data model since an object may be decomposed (fragmented) more than one alternative. The segments are to deal with one decomposing alternative, while the atoms are to support all alternatives. Referring to the image example above, the collection of segments could be based on either color component or quality degree, while the collection of atoms would be based on the combination of the two. The other reason of defining segments is to balance fine granularity and fast modulation. The granularity of atoms is very small. Performing modulation on atoms could take a longer time than that on segments.

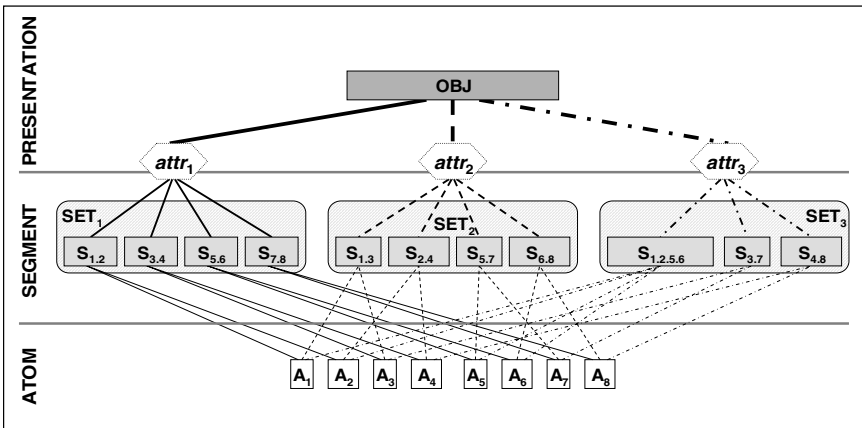


Fig. 1. Layers of Data

Consequently, the data model comprises three data layers: presentation, segment, and atom. The presentation layer is where an object and its representations (variants) fall into. Figure 1 illustrates the organization of the data layers. Entities in each data layer have specific characteristics. An entity of the presentation layer (in the form of

an object) is heterogeneous since it contains fragments with diverse attribute values. As indicated by the data layer's name, the entity is presentable. On the contrary, an entity of the segment layer (in the form of a segment) is unpresentable. However, homogeneity starts to appear on the segment-layer entity; that is, the segment's fragments support one or more similar attribute values (some attributes still vary in value). An entity of the atom layer (in the form of an atom) is homogeneous and has a unique combination of attribute values. Clearly, the atom-layer entity is very primitive and far from presentable. In general, the entity's granularity becomes finer and the entity's fragments become more homogeneous as we go from the presentation layer to the atom layer.

In the data model (see Figure 1), an object (OBJ) can be decomposed into segments $S_{1,2}$, $S_{3,4}$, $S_{5,6}$, and $S_{7,8}$ based on a set of attributes $attr_1$. Further, the segments can be divided into atoms $A_1, A_2, A_3, A_4, A_5, A_6, A_7,$ and A_8 . These are two-way transformations. We can regard them as top-down transformations (from the presentation layer to the atom layer). Conversely, we can also view them as bottom-up transformations (from the atom layer to the presentation layer). Hence, we can say that segments $S_{1,2}$, $S_{3,4}$, $S_{5,6}$, and $S_{7,8}$ are constructed by collecting the atoms with the same values for the set of attributes $attr_1$. Since there may be more than one decomposing alternative, different sets of segments may be produced. Of the object in the illustration, three sets of segments – SET_1 , SET_2 , and SET_3 – can be formed, each of which is associated with a different set of attributes – $attr_1$, $attr_2$, and $attr_3$, respectively.

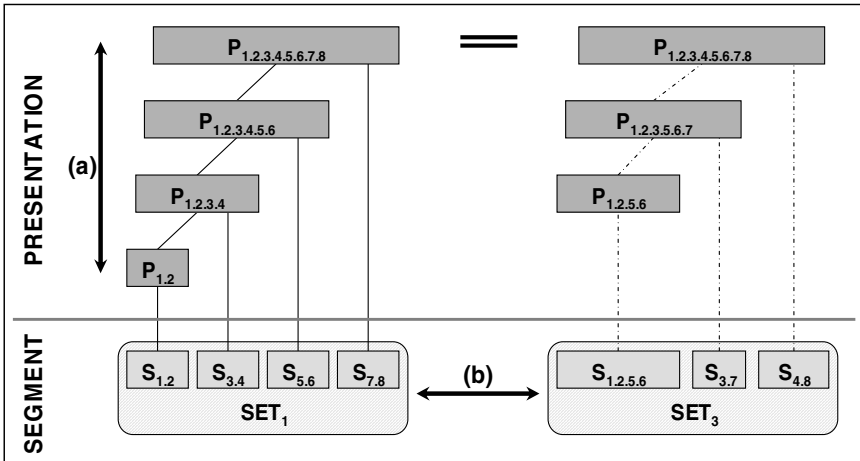


Fig. 2. Generating Object's Representations Modulating Aspects: (a) Scalability, and (b) Variability

Continuing the previous illustration, Figure 2 shows how various object's representations are constructed. An object's representation is constructed from one or more segments. For example (based on SET_1), representation $P_{1,2}$ is made up of segments $S_{1,2}$, while representation $P_{1,2,3,4}$ is made up of segments $S_{1,2}$ and $S_{3,4}$. Note that segment $S_{1,2}$ itself is unpresentable. Some headers (metadata) have to be added to

segment $S_{1,2}$ to become representation $P_{1,2}$. Adding one or more segments to a representation can also result in a more sophisticated representation (e.g., representation $P_{1,2,3,4,5,6}$ is constructed from representation $P_{1,2,3,4}$ and segment $S_{5,6}$). Likewise, eliminating one or more segments from a representation can result in a less sophisticated representation (e.g., removing segment $S_{3,4}$ from representation $P_{1,2,3,4}$ yields representation $P_{1,2}$). When we combine the entire set of segments, the yielded representation is indeed the original object. Besides manipulating segments of one set (in this case, SET_1), we can also work on segments of the other set (SET_3 , for example) to gather the object's representations, exploiting the other properties of the object. Referring back to the image example at the beginning of this subsection, we could generate varieties of representations by exploiting the color component as well as the quality degree.

3.2.2 Modulating Processes

Processes in modulation are quite straightforward, and closely related to the data model. We begin by introducing two modulating aspects (refer to Figure 2): *scalability* and *variability*. As explained in the data model, an object may be decomposed in more than one alternative, using different sets of attributes and resulting in different sets of segments. The number of sets determines the number of modulating alternatives. The variability aspect of modulation (horizontal dimension in Figure 2) deals with the modulating alternatives (variations) supported by a representation. Within a particular set, the segments can construct representations with different degrees. Depending on the attributes exploited by the set, the representations can have different degrees in quality, or resolution, or the number of color components, or anything else. The scalability aspect of modulation (vertical dimension in Figure 2) deals with the degree (scale) retained by a representation with respect to a particular variation. In other words, for every representation we can get by modulating an object, we can determine specifically what the variations are (i.e., attributes that can be exploited) and what the scale of each variation is (i.e., the degree of that variation). For example, an image's representation has quality as its variation, with 7 out of 10 as the quality scale. Note that several variations – with several corresponding scales – may be attached to an object's representation.

Modulating processes are performed on an object along the two modulating aspects. Consequently, there are two types of modulating processes. The first type, which consists of scaling processes, is to decrease or increase the number of segments in a representation to produce another representation of the same set of segments. It exploits the scalability aspect of an object. The processes to be performed depend on the target variation (i.e., the set of segments to be employed). Such an example is a modulating process to reduce the quality of an image. The second type of processes, which includes many varying processes, is to change the target variation, so that a representation can be transformed into another representation of a different set of segments. It exploits the variability aspect of an object. Remind that changing the target variation causes rearrangement of atoms to a different set of segments. A varying process should not work alone, thus it must work together with some scaling processes. In addition, it is usually applied to a representation rather than a full object. For instance, transforming a low quality, color image into a high quality, grayscale image involves a varying process (to change the target variation from quality to color

component) and two scaling processes (to decrease the number of color components and to increase the quality).

Depending on the object's data format, some modulating processes may perform concurrently. Using the last example, the low quality, color image can be transformed first to a low quality, grayscale image (changing the target variation while reducing the number of color components), and then transformed again to a high quality, grayscale image (adding the intersection of high quality and grayscale segments). If the modulating processes cannot perform concurrently, we have to transform the low quality, color image to a high quality, color image (adding the quality segments). After that, we change the target variation (from quality to color component) and reduce the number of color components, resulting in a high quality, grayscale image. The latter method is less efficient than the former since a larger amount of data is wasted.

3.2.3 Metadata

The modulating processes cannot modulate an object properly if they do not know what variations are supported by the object and what the current scale is for each variation. All of this supported information should accompany the object. The role of metadata (the common term to label data describing the main data) is quite important in modulation. From modulation's point of view, there are three types of metadata, which are useful for fulfilling the modulation's goals. Those three types of metadata are as follows:

- Direct support: information directly related to the modulating processes. It includes information about the variations supported by a representation and the scales of each variation, as well as information about the atoms of the representation (i.e., attributes, positions, sizes, etc.).
- Indirect support: information not directly related to the modulating processes but quite helpful to a decision maker. The information describes the representation to be generated. Such information is estimated size, estimated dimensions, and so forth. A decision maker (a client or a proxy) often needs to compare several representations before selecting the most appropriate one. The indirect support could be one of the determinants in the decision making process.
- Hints/instructions: guidance from the server or other network elements, also helpful to a decision maker. This information is nothing to do with modulation, but it can influence the decision making process. It includes available network bandwidth, caching capability (of a proxy), and so forth.

Some metadata is embedded on the corresponding object, but some has to be provided separately. Direct support and some indirect support are often attached to the object, either on the object's header or scattered in the object, depending on the data format. If the metadata is not available in the object, it can be provided in a special metadata file, in HTML tags, or in HTTP headers. Some metadata is available explicitly, whereas some is implicit and has to be derived from other information. Most direct support is explicit information, while indirect support and hints sometimes have to be derived from, calculated, or estimated from other information.

3.3 Data Format's Requirements

So far, we have elaborated the concept of modulation in details, involving its characteristics, conceptual framework, and formulation. Now let us assess how to materialize this concept in the real world. As the data model is the essential element of modulation and the data model is related to data formats in the real world, we need to examine what kind of data format can support modulation. Firstly, the data format must support scalable presentation, which modulation can exploit. There is no use to apply modulation to a data format if it only supports single presentation. The progressive multimedia formats are often scalable, so they are potential candidates for modulation. Secondly, the scalable presentation specified by the data format should be constructed of fragments, which can be easily identified in and fetched from the data. Lastly, exploiting the scalable presentation, a representation can be built without recoding (decoding and encoding) the data. Apart from the first condition, the specification for the modulation's data format may be loosened here and there.

4 Conclusion

In this paper, we have described a new transcoding approach called modulation. Different from transcoding, which involves a lot of complex factors, modulation is straightforward and fast. Basically, it is an eliminating and/or adding process applied to the building blocks (fragments) of an object. We have outlined the conceptual framework of modulation. The most important requirement for modulation is the scalable presentation supported by the object's data format. Due to its potential benefits and observing the trend of object fragmentation presently, we believe that modulation can proliferate broadly in many types of web objects.

References

- [1] S. Chandra and C. S. Ellis. JPEG Compression Metric as a Quality Aware Image Transcoding. In *Proc. of 2nd USENIX Symposium on Internet Technologies and Systems (USITS)*, Boulder (CO), October 1999.
- [2] C. H. Chi and Y. Cao. Pervasive Web Content Delivery with Efficient Data Reuse. In *Proc. of 7th International Workshop on Web Content Caching and Distribution (WCW)*, Boulder (CO), August 2002.
- [3] C. Christopoulos, A. Skodras, and T. Ebrahimi. The JPEG2000 Still Image Coding System: An Overview. *IEEE Transactions on Consumer Electronics*, vol. 46, no. 4, pp. 1103–1127, November 2000.
- [4] Fox, S. D. Gribble, E. A. Brewer, and E. Amir. Adapting to Network and Client Variability via On-Demand Dynamic Distillation. In *Proc. of 7th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, October 1996.
- [5] S. Deshpande and W. Zeng. Scalable Streaming of JPEG2000 Images using Hypertext Transfer Protocol. In *Proc. of 9th ACM Multimedia Conference*, Canada, October 2001.
- [6] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas. Dynamic Adaptation in an Image Transcoding Proxy for Mobile Web Browsing. *IEEE Personal Communications*, vol. 5, no. 6, pp. 8–17, December 1998.
- [7] The JPEG2000 Official Homepage. <http://www.jpeg.org/JPEG2000.htm>.

- [8] B. Knutsson, H. H. Lu, and J. Mogul. Architecture and Pragmatics of Server-Directed Transcoding. In *Proc. of 7th International Workshop on Web Content Caching and Distribution (WCW)*, August 2002.
- [9] Maheshwari, A. Sharma, K. Ramamritham, and P. Shenoy. TranSquid: Transcoding and Caching Proxy for Heterogeneous E-Commerce Environments. In *Proc. of 12th International Workshop on Research Issues in Data Engineering (RIDE)*, San Jose (CA), February 2002.
- [10] J. C. Mogul. Server-Directed Transcoding. *Computer Communications*, vol. 24, no. 2, pp. 155–162, February 2001.
- [11] R. Mohan, J. R. Smith, and C. S. Li. Adapting Multimedia Internet Content for Universal Access. *IEEE Transactions on Multimedia*, vol. 1, no. 1, pp. 104–114, March 1999.
- [12] D. Taubman. Remote Browsing of JPEG2000 Images. In *Proc. of IEEE International Conference on Image Processing (ICIP)*, Rochester (NY), September 2002.

A Deadline-Sensitive Approach for Real-Time Processing of Sliding Windows¹

Wu Shanshan, Yu Ge, Yu Yaxin, Ou Zhengyu, Yang Xinhua, and Gu Yu

Northeastern University, Shenyang, China 110004
wssmail@126.com

Abstract. In a growing number of information processing applications, data takes the form of continuous data streams rather than traditional stored databases. Most of these applications have sophisticated real-time constraint that needs to be met under unbounded, high-volume, time-varying data streams. We introduce *deadline* as a real-time constraint of continuous query over data streams. Specifically, a deadline-sensitive approach for sliding window processing is proposed, which predicts a sliding window would satisfy its *deadline* or not. Once deadline missing is predicted, *HopDrop*, a proposed load shedding strategy would be applied to lighten the system burden in advance. Furthermore, feedback control mechanism is applied to improve the adaptivity of our approach. Extensive experimental results are presented and analyzed to validate our strategies.

1 Introduction

Data streams have emerged as an important paradigm for processing data that arrives and needs to be processed continuously. For instance, military applications that monitor readings from sensors worn by soldiers, financial analysis applications that monitor streams of stock data reported from various stock exchanges, temperature monitoring in chemical reactions and so on. In most of these applications, data is generated at distributed sources, and then streamed to a central server where a Data Stream Management System (DSMS) [1, 2, 3, 4] is running. Continuous queries from the applications are registered to DSMS and are evaluated over the incoming data continuously.

A salient feature of continuous queries injected to DSMS is the sophisticated real-time constraint, for the streams are usually short-lived and to be immediately consumed. Therefore, unlike a traditional DBMS where people are mostly interested in the correctness of query results, queries in a DSMS generally require to be delivered in a timely fashion. Longer delay usually means poorer data freshness and higher uncertainty, and sometimes the delay exceeding certain threshold value may incur the loss of profit or product quality, due to possible changes in the market or manufacturing status catastrophic results. Therefore, we formally put forward the term, *deadline*, as real-time constraint of data stream processing. We associate

¹ Supported by National Science Foundation (60473073) and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of the Ministry of Education.

deadline with continuous queries in DSMS as the maximum response delay tolerated by the applications. Aiming at the real-time constraint of stream-based applications, we address the problem of deadline-sensitive data stream processing in this paper.

Since streams are potentially unbounded in size, computation over the entire streams seems not practical. And for many stream-based applications, recent elements of a stream are more important than those arrived long ago. This preference for recent elements is commonly expressed using a sliding window [5, 6], which identifies a portion of the stream that arrived between "now" and some recent time in the past. So this paper focuses on a deadline-sensitive approach for processing of sliding window.

When processing high-volume, bursty data streams, even though the average arrival rate is within computational limits, there may be bursts of high load leading to accumulation of input data to be processed, due to finite CPU and memory capacity. As a result, latency will deteriorate, the number of data missing their deadline will increase, and the overall system throughput will decrease accordingly. Therefore, it is particularly important for DSMS to be able to automatically adapt to unanticipated bursts in input data rates. In addition, an overloaded system will be unable to process all of its input data to keep up with the rate of data arrival, so load shedding, i.e., discarding some fraction of the unprocessed data, becomes necessary in order for the system to continue to provide up-to-date response. In our deadline-sensitive approach for real-time processing of sliding windows, when the data rate fluctuates, chances are that the system may predict the arrived windows could satisfy their deadline or not. If deadline missing is predicted, load shed could be applied in advance to relieve the system. In summary, we make the following contributions in this paper:

- We summarize the real-time constraint of stream processing, and formally put forward the term *deadline* for real-time processing of data streams. Particularly, we focus on deadline-sensitive sliding window modeling and processing.
- We introduce a *H-D-P model* to analyze adaptativity of sliding window based processing to data rate fluctuation, based on which a *DSAT prediction strategy* is proposed for real-time processing over data stream sliding window.
- A deadline-sensitive processing architecture with feedback control is proposed, in which *DSAT* prediction is applied. Once deadline missing is predicted, a proposed load shedding strategy, *HopDrop*, is applied to relieve the system in advance.

The rest of this paper is organized as follows. We begin in Section 2 by formalizing the problem and establishing notations and models. Section 3 presents our deadline-sensitive approach for real-time processing of sliding windows. Experimental results are presented in Section 4. Section 5 discusses the related work. Finally, we end the paper with conclusions and future work in Section 6.

2 Preliminaries

In this section, we introduce some key concepts and notation. Without loss of generality, we assume that the timestamp referred in this paper is defined in a nonnegative integer domain.

2.1 Definition of Deadline

In an ideal case, at any point of time t , the answers resulting from the input at time t would be available instantaneously. We refer to t in this case as the *output create-time*, denoted by t_{create} . However, real systems produce the output with certain delay. We refer to the output time in this case as the *output release-time*, denoted by $t_{release}$. And the real-time constraint of a query is to minimize the response delay (i.e., $t_{release} - t_{create}$) within certain threshold value.

Definition 1: *deadline*. Deadline of a continuous query Q is the maximum response delay tolerated by the application, denoted by D_Q . For any output of Q :

- $t_{release} - t_{create} \leq D_Q$. Represents that the answer satisfies its deadline, and it is *DSAT*;
- $t_{release} - t_{create} > D_Q$. Represents that the answer misses its deadline, and it is *DMISS*.

For stream-based applications, some are with soft deadline semantics, in which tuples that have already missed their deadlines will not all be discarded though less useful. Contrarily, some are with hard deadline semantics, in which tuples that have already missed their deadlines will be no use any more so that to be discarded. Hard deadline semantics are common in many real-time applications.

The *deadline satisfy ratio (DSR)* is one of the most important performance metrics in real-time applications, and we give definition to *deadline satisfy ratio* in real-time data stream processing as in Equation (1), where $\#DMISS$ and $\#DSAT$ represent the number of tuples that have missed and met their deadlines, respectively.

$$DSR = 100 \times \frac{\#DSAT}{\#DSAT + \#DMISS} (\%) \tag{1}$$

2.2 Deadline-Sensitive Sliding Window Model

A sliding window over a stream always contains a bag of the most recent tuples seen so far in the stream. And the window slides over the infinite continuous data stream with certain frequency termed as hop [7]. There are two ways to build up windows: time-based window and count-based window [8]. Here we focus on the time-based window, and it can be easily applied to the tuple-based window.

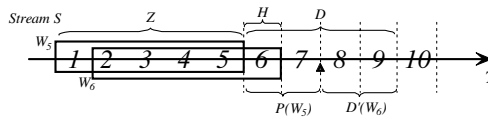


Fig. 1. Illustration of *deadline-sensitive sliding window*

As illustrated in Figure 1, window W with size Z and hop H consists of tuples arrived in Stream S in recent Z time units, and slides forward every H time units, assuming $0 < H \leq Z$. At each point of time t , the new arrived tuples are inserted into the window, and the expired tuples are deleted from the window. Thus, a new *window instance* at time t is built up, denoted by W_t . As far as W_t is concerned, the *output create-time* and *output release-time* are denoted by $t_{create}(W_t)$ and $t_{release}(W_t)$, respectively. For each window instance W_t :

- $t_{release}(W_t) - t_{create}(W_t) \leq D_Q$, W_t is *DSAT*;
- $t_{release}(W_t) - t_{create}(W_t) > D_Q$, W_t is *DMASS*.

In this case, #*DSAT* and #*DMASS* in the definition of *DSR* given in Equation (1) represent the number of window instances that meet and miss deadlines, respectively.

In our sliding window processing model, we employ the “incremental evaluation” philosophy. Compared with W_{t-H} , the data arrived in recent H time units, i.e., $W_t - W_{t-H}$ are inserted into W_t , and the oldest data of H time units in W_{t-H} , i.e., $W_{t-H} - W_t$, are added. Therefore, as shown in Equation (2), the output of W_t , denoted by $Output(W_t)$, can be achieved by adding the effect of inserted data and that of deleted data over the results of W_{t-H} , denoted by $Output^+(W_t - W_{t-H})$ and $Output^-(W_{t-H} - W_t)$ respectively. Noted that both insertion and deletion over a window may incur addition and/or invalidation to overall output [9], so $Output^+(\cdot)$ and $Output^-(\cdot)$ may add tuples to the overall output or delete tuples from overall output.

$$Output(W_t) = Output(W_{t-H}) + Output^+(W_t - W_{t-H}) + Output^-(W_{t-H} - W_t), t = Z + i \cdot H, i = 1, 2, \dots \quad (2)$$

For the cost of each window instance W_t , besides the *window maintenance cost* P_m , the computation cost $P(W_t)$ consists of the cost of adding the effect of increased data and that of deleting the effect of decreased data in the window, denoted by $P^+(W_t - W_{t-H})$ and $P^-(W_{t-H} - W_t)$ respectively. So the *computation cost model* of a window instance is shown as following in Equation (3).

$$P(W_t) = P^+(W_t - W_{t-H}) + P^-(W_{t-H} - W_t) + P_m, t = Z + i \cdot H, i = 1, 2, \dots \quad (3)$$

For queries over deadline-sensitive sliding windows, the absolute goal is to minimize the processing cost of each window instance to meet its deadline. (Of course, *waiting time* is another cause of response delay, which does not belong to the computation cost model discussed in this paper.)

2.3 D-H-P Model

For a sliding window W , a three-way relationship among hop, deadline and processing cost is called a *H-D-P* model.

P-H Relation. To keep up with window updating, processing system should make effort to finish processing each window in H time units, or windows will pile up and the system will become too overloaded to hold in memory or to satisfy deadline. Accordingly, an *overload-factor* is introduced to describe overload effect.

Definition 2: *overload-factor* of W_t , $O(W_t)$, is defined in Equation (4).

$$O(W_t) = \frac{P(W_t) - H}{H} \quad \text{where } t \geq Z \quad (4)$$

With the assumption that W_t is *DSAT*:

- $O(W_t) \leq 0$. W_t finishes computation before W_{t+H} is built up, which makes a negative overload burden on the processing;
- $O(W_t) > 0$. W_{t+H} has built up when W_t finishes computation, which makes a positive overload burden to the processing. In other words, the processing of W_t takes up some time of subsequent windows.

Definition 3: *accumulated overload-factor* from W_t to W_s , AO_{ts} . As given in Equation (5), AO_{ts} is defined as the accumulation of O_i of the window instances from W_t to W_s .

$$AO_{ts} = \sum_{i=t}^s O_i \text{ where } s \geq t \geq Z \tag{5}$$

Definition 4: *balance point*. *Balance point* is given definition in recursion as follows.

- 1) Time point Z is the first balance point;
- 2) If time point t is a balance point, then time point s is a balance point if and only if $AO_{ts} \leq 0 \wedge \neg \exists l (AO_{ts-H} \leq 0 \wedge Z \leq t \leq l < s)$. In this case, t is called the *balance point* of the round of overload-factor accumulation.

Known from Definition 2, 3 and 4, at a balance point t , W_{t-H} has finished computation and W_t has been built up. So that the accumulated overload burden makes no effect on W_t and the subsequent windows. Then a new round of overload-factor accumulation begins from *balance point* t .

P-D Relation. Considering the accumulated overload effect, *deadline* of W_t , i.e., $D(W_t)$, should to be modified by $D(W_t) = D - H * AO_{bt-H}$, where b is the *balance point* of current round of overload-factor accumulation. Here, we introduce a *DSAT-factor* deprived from P-D relation.

Definition 5: *DSAT-factor* of W_t , $\delta(W_t)$ is defined in Equation (6).

$$\delta(W_t) = \frac{P(W_t)}{D(W_t)} = \frac{P(W_t)}{D - H * AO_{bt-H}}, \text{ where } t \geq Z \tag{6}$$

- $0 < \delta(W_t) \leq 1$. W_t is *DSAT*;
- $\delta(W_t) > 1$. W_t is *DMISS*.

Moreover, in the range of $(0,1]$, the closer $\delta(W_t)$ is to 1, the more urgent deadline is. On the contrary, the closer $\delta(W_t)$ is to 0, the slacker deadline is.

D-H Relation. D-H relation is studied in the following two aspects:

- $D \leq H$. In this case, deadline constraint restricts window computation to keep up with window updating, or even ahead of window updating. Therefore, time point that each window built up is a balance point.
- $D > H$. In this case, slacker deadline constraint allows window computation to lag behind window updating. Under this circumstance, *H-D-P* model presents a kind of adaptivity to the fluctuation of data rate among consecutive window instances. There are three possible scenarios for each window instance W_t :

Given that $D > H$,

- 1) $P(W_t) > D(W_t) > H$. W_t is *DMISS*;
- 2) $D(W_t) \geq P(W_t) > H$, i.e., $O(W_t) > 0$, W_t is *DSAT*, and it aggravate the accumulated overload effect of successive processing;
- 3) $D(W_t) > H \geq P(W_t)$, i.e., $O(W_t) \leq 0$, W_t is *DSAT*, and it can lighten the accumulated overload effect of successive processing.

As a conclusion, since $O(W_t)$ fluctuates between “+” and “-”, negative overload effect and positive overload effect can compensate each other during accumulation. Hence, when the bursts of high arrival rate have receded, the piled unprocessed windows have the possible to be cleared up. In other words, processing time under

light load may compensate for that under heavy load, which is a kind of adaptivity to the fluctuation of data rate in $H-D-P$ model.

3 Deadline-Sensitive Processing Approach

In this section, we propose a novel deadline-sensitive approach for processing of sliding windows. We give a deadline-sensitive processing architecture with feedback control, and present the *Deadline-SAT prediction* and *HopDrop* strategy.

3.1 Deadline-Sensitive Processing Architecture with Feedback Control

Here we give an overview of our deadline-sensitive processing architecture. The general outline of the architecture is given in Figure 2.

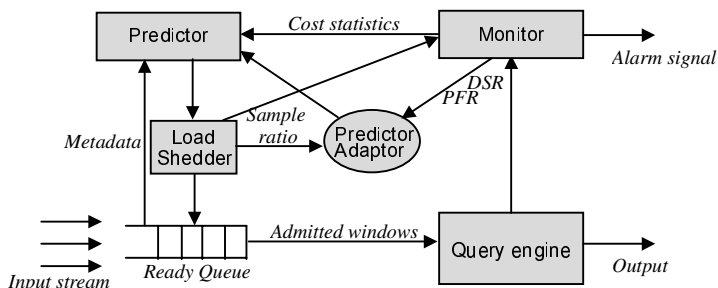


Fig. 2. Deadline-Sensitive Processing Architecture with Feedback Control

- 1) Arrived windows are placed in the *ready queue*.
- 2) The *query engine* continuously processes the admitted input windows and delivers corresponding output to applications outside the system.
- 3) The *monitor* measures *DSR* and predict failure ratio (*PFR*), collects statistics of processing cost and forwards the information to the *predictor*.
- 4) Based on the metadata of arrived input and information from the monitor, the *predictor* estimates the newly arrived windows are *DSAT* or not (Section 3.2.)
- 5) If a window is predicted *DMISS*, load shedding will be applied by *load shedder*.
- 6) To provide robustness and adaptivity to the processing architecture, *predictor adaptor* periodically adjusts the *predictor* according to the feedback *PFR* (Predict Failure Ratio), *DSR* and sample ratio.
- 7) Moreover, if *DSR* or sample rate deteriorate to the extreme value, alarm informs the application outside that the load is too overloaded to be held in the system.

3.2 Deadline-SAT Prediction Strategy

In the case of $D > H$, W_t is built up at time point t , and it has to finish computation within D time units. At any instant τ during the computation of W_t , there exists an exclusive k ($k \geq 0$) that satisfies $k * H \leq \tau - t < (k+1) * H$, where $\tau \in [t, t+D)$. In other words, subsequent k windows would have arrived during the computation of W_t . Chances are

that the system could predict whether the k windows are *DSAT* or not. Take the sliding window illustrated in Figure 1 as an example. W_5 finishes computation at 7^{th} time unit, and it is obvious that we can predict W_6 and W_7 . If W_6 is predicted *DMISS*, load shedding would be applied over it in advance.

Look at the *computation cost model* of a window instance given in Equation (3), P_m closes to a constant value that can be easily collected at the startup phase of the system. As far as $P^+(W_t-W_{t-H})$ and $P^-(W_{t-H}-W_t)$ are concerned, there are three leading factors: (1) the number of newly arrived tuples and that of expired tuples, which are denoted by $|W_t-W_{t-H}|$ and $|W_{t-H}-W_t|$, respectively. (2) The operations of the query plan. Based on whether the processing cost is correlated to the window size Z or not, the operations can be simply classified into two categories: Operations whose processing cost is unrelated to window size, such as SUM, COUNT, AVERAGE, etc.; and the operations whose processing cost is somehow correlated to window size, such as Window Join, MAX, MIN, DISTINCT, etc. (3) Physical implementation policy of the operations. However, once the query plan and implementation policy are determined, (2) and (3) can be reflected by the average processing cost of new arrived tuples and the expired tuples. Accordingly, during the runtime, the *monitor* periodically collects the average processing cost of new arrived tuples and the expired tuples, denoted by P^+_0 and P^-_0 , and forwards the information to the *predictor*. Based on the metadata of arrived input and information from the monitor, the *predictor* estimates the processing cost of window instance W_t , i.e., $\hat{P}(W_t)$, using *window cost predict model* given in Equation (7), where the *cost adjust factor* ρ is initially set to 1.

$$\hat{P}(W_t) = (P^+_0 * |W_t - W_{t-H}| + P^-_0 * |W_{t-H} - W_t|) * \rho + P_m, t = Z + i \cdot H, i = 1, 2, \dots \tag{7}$$

Thus, we can evaluate the estimated values of $O(W_t)$, $D(W_t)$, and $\delta(W_t)$, denoted by $\hat{O}(W_t)$, $\hat{D}(W_t)$ and $\hat{\delta}(W_t)$ respectively, in order to identify the window under prediction is *DSAT* or not according to Definition (5). Detailed algorithm is shown in Table 1.

Table 1. DSAT Prediction Algorithm

1. while ($\tau < D(W_t) \wedge W_t$ not finished)	10.	if ($\hat{\delta}(W_k) > 1$)
2. {	11.	{
3. Process window W_t ;	12.	Predict W_k is <i>DMISS</i> ;
4. if (A new win W_k builds up)	13.	Shed load;
5. {	14.	}
6. $\hat{P}(W_t) = (P^+_0 * W_t - W_{t-H} + P^-_0 * W_{t-H} - W_t) * \rho + P_m$;	15.	else
7. $\hat{O}(W_t) = \frac{\hat{P}(W_t) - H}{H}$;	16.	Predict new W_k is <i>DSAT</i> ;
8. $\hat{D}(W_t) = D - H * \left(\sum_{j=b}^{t-1} O(W_j) + \sum_{j=t}^{k-1} \hat{O}(W_j) \right)$;	17.	} //endif new W_k builds up
9. $\hat{\delta}(W_t) = \frac{\hat{P}(W_t)}{\hat{D}(W_t)}$;	18.	} //endwhile
	19.	if ($\tau > D(W_t)$)
	20.	Report prediction failure;
	21.	Make prediction adaptation;
	22.	if (W_t finished $\wedge AO_{t-1} \leq 0$)
	23.	Set new balance point t ;

To provide robustness and adaptivity to our *DSAT* prediction strategy, we apply feedback-control based *prediction adaptation*. As far as the estimated cost of window instance is concerned, it is evaluated based on the runtime statistics and input

metadata. And the runtime statistics is where the predict error may come from mostly. When the cost of window instance is overestimated, load shedding would be applied on the windows that might not miss their deadlines. In this case, both *DSR* and the *sample ratio* of the load shedder would decrease hardly. On the contrary, when the cost of window instances is underestimated, windows would prefer to miss their deadline though they are predicted *DSAT*. In this case, *DSR* would deteriorate, and *PFR* would increase remarkably. Accordingly, when *DSR* deteriorates we make slight adjustment to the *cost adjust factor* ρ with respect to sample ratio and *DMR*.

3.3 A Load Shedding Strategy: *HopDrop*

Once a window is predicted *DMISS*, load shedding should be applied over it to lighten the system load in advance without loss of output performance.

In our “incremental evaluation” philosophy, load shedding can be customized as illustrated in Figure 3. When W_t is predicted *DMISS*, the new arrived tuples in W_t , i.e., $W_t - W_{t-H}$, are discarded without processing. And the cost of W_t can be diminished to:

$$P'(W_t) = P \cdot (W_{t-H} - W_t) + P_m, t = Z + i \cdot H, i = 1, 2 \dots$$

Compared with Equation (3), the load shedding of W_t saves system time as much as $P \cdot (W_t - W_{t-H})$, which can

be used to process subsequent windows. And there is no additional work for the posterior window W_{t+H} to do. As it is shown above, the load shedding strategy used in our deadline-sensitive approach is just to skip the new arrived hop, so we term it as *HopDrop*.

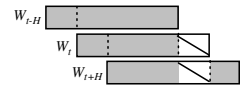


Fig. 3. *HopDrop* illustration

Noted that after applying *HopDrop* over W_t , if W_{t+H} has not built up, we can buffer the tuples to be dropped and process them as much as possible within their deadline; and if the load of W_{t+H} is relatively small, the buffered tuples could also be processed with best effort within their deadline. Moreover, load-shedding strategy used in *DSAT* prediction algorithm is not limit to *HopDrop* merely.

4 Experimental Results

We begin with a brief description of our simulation framework in Section 4.1. The experiment results and analysis are presented in Section 4.2.

4.1 Experiment Setup

The Internet Traffic Archive [10] is a good source of real-world stream data sets. One of their races, named “*LBL-PKT-4*”, includes 863,000 records representing one hour’s worth of all wide-area traffic between Berkeley Lab and the rest of the world. We use this trace as the data set since it can provide a kind of natural and authentic stream rate fluctuation. Each record contains one timestamp column and five integer valued columns, *srcHost*, *destHost*, *srcPort*, *destPort*, and *size*.

The sliding windows in the queries were based on the timestamps presented in the data items. Processing cost of each packet is randomized in a given region. To play fair comparisons, all the experiments are conducted with the same materialized randomization. Once windows are predicted *DMISS*, *HopDrop* is applied to make

load shedding. In order to evaluate the performance of our deadline-sensitive approach (*DSA*), we also implemented the best-effort algorithm (*BEA*) in the same experiment environment. In *BEA*, if a window finishes computation within deadline, a *DSAT* window is contributed. Otherwise, if it does not finish computation till deadline, the unfinished tuples would be discarded.

Furthermore, it is known from Section 3.2 that the window processing cost is somehow related to the operation over the sliding window. Accordingly, we conduct two groups of experiments. In Group (a), operation cost of window instance is unrelated to the window size. We ask for the average network traffic from different IP addresses. In Group (b), operation cost of window instance is related to the window size. It is asked to return the IP address that made the max network traffic within a time period specified by window size. We vary the window size, hop and deadline for each group of experiment to see about *DSR* and *PFR* under different circumstances.

4.2 Experiment Analysis

Firstly, we study our analytical results regarding the *H-D-P* model discussed in Section 2.3. According to the definition of *DSAT factor* in Equation (6), we discuss different factors that have effects on *DSR*. (1) Window size *Z*. For the operation whose processing cost is unrelated to window size, window size has no effect on the value of $\hat{\alpha}(W_i)$. Subsequently, *DSR* of such operation is not affected by the variation of window size, as illustrated in Figure 4(a). And for the operation whose processing cost is related to window size, $P(W_i)$ increases with window size. And then $\hat{\alpha}(W_i)$ increases accordingly, which means that windows are easier to miss deadline. This is the reason why the *DSR* in Figure 4(b) decreases when window size scales up. (2) Window hop *H*. Equation (3) and (7) tell that if *H* increases, $P(W_i)$ will increase accordingly. Therefore, $\hat{\alpha}(W_i)$ will increase, and windows prefer to miss deadline. As a result, *DSR* in both Figure 5(a) and Figure 5(b) deteriorate as window hop increases. (3) Deadline *D*. Obviously, longer deadline provides slacker real-time constraint, and more windows will satisfy their deadline as shown in Figure 6(a) and Figure 6(b). Hereby, the experimental results validate the correctness of our *H-D-P* model.

Secondly, *PFR* in Figure 4-6 are quite low, which shows the precision of our *DSAT* predict strategy. As addressed in *DSAT* prediction algorithm (Section 3.2), when estimating processing cost of window instances, errors would be involved. Assuming the error of estimated processing cost of a window as ξ , we have:

$$\hat{\delta}(W_i) = \frac{\hat{P}(W_i)}{D(W_i)} = \frac{P(W_i) + \xi}{D - \sum_{j=b}^{i-H} (P(W_j) - H)}$$

Therefore, the error of $\hat{\delta}(W_i)$ resulting from ξ is magnified to:

$$\text{error of } \hat{\delta}(W_i) = \frac{\xi}{D - \sum_{j=b}^{i-H} (P(W_j) - H)}$$

Accordingly, it is quite easy to explain the *PFR* curves of *DSA* in Figure 4-6. For the operation whose cost is not affected by window size, the error is also unrelated to the variation of window size, while for the operation whose cost is related to window size, the error is magnified as $P(W_i)$ increases. When window hop increases, however,

$P(W_i)$ increases accordingly, so the variation of $\delta(W_i)$ is not determined. It is obvious that the increase of deadline diminishes the error, so that PFR decreases as a result. From the other point of view, the magnified error is somehow in proportion to $DSAT$. Therefore, though PFR may increase with the variation of window size, more failures occur only when DSR is lower, which makes feeble influence to most applications.

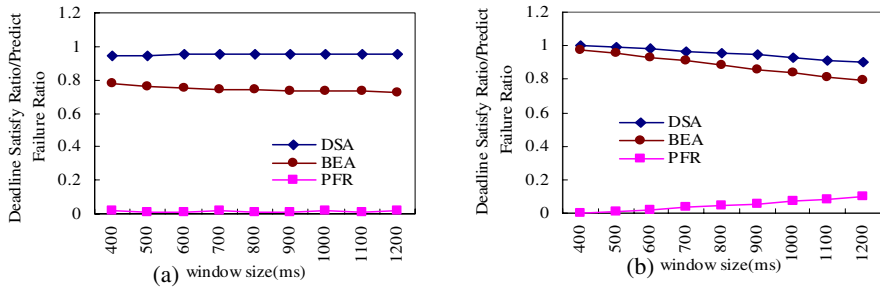


Fig. 4. Effect of Window Size (HOP=300ms Deadline=500ms)

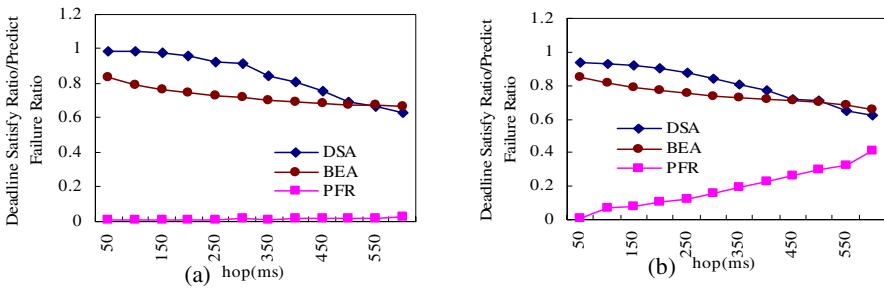


Fig. 5. Effect of Window Hop (window size=1500ms Deadline=500ms)

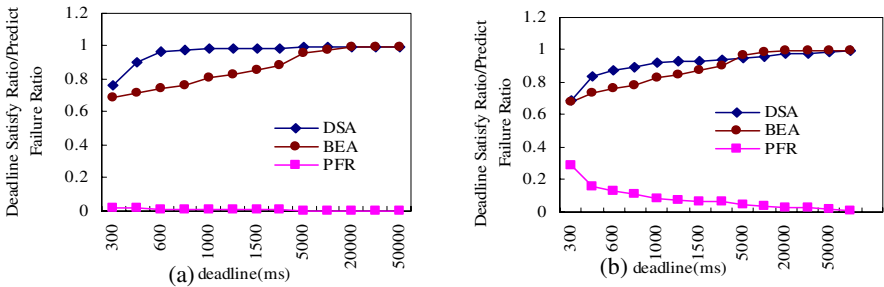


Fig. 6. Effect of Deadline (Window size=1500ms HOP=300ms)

Another performance concerned is the boost of DSR won by DSA over BEA . We are pleased to see the DSR of DSA outperforms that of BEA almost all along. When predict chances are more, more $DIMSS$ window can be discarded in advance, so that time can be saved to compute more $DSAT$ windows. In a word, more predict chances

may bring improvement of *DSR*, and vice versa. As for window size, it has no effect on the chances of prediction, which is coincided by Figure 4. As for Figure 5-6, since our prediction strategy is only practicable in the case of $D \geq H$, when D draws near to H or even below H , chances of prediction come down, so *DSA* provides less benefit or even the same performance as *BEA*. On the other side of the spectrum, when *DSR* of both algorithms improve, they all draw towards the ideal case, *i.e.*, $DSR=1$.

5 Related Work

Recently, there has been considerable research activity pertaining to stream systems and data stream algorithms. Some systems that incorporate stream processing include Aurora [4, 11], Niagara [1], STREAM [2], and Telegraph [3], among many others.

Stream-based applications require answers of continuous queries should be delivered in a timely fashion. Real-time databases, which have real-time constraint on their transactions [12, 13], associate deadline with individual transactions. Transactions that cannot be finished within deadline would make no value or even negative value to the system. Here we associate *deadline* with each continuous query over streams. To the best of our knowledge, most of the literature on real-time data stream processing is to make best effort to deliver answers with minimum average response delay [14, 15, 16, 17]. However, *deadline* constraint of continuous query has not been definitely termed in data stream processing, and little attention has been paid to deadline sensitive adaptive processing or deadline satisfaction prediction.

Another relevant research area deals with load shedding. There are two main approaches of load shedding. The first relies on random load shedding, *i.e.*, tuples are removed based on arrival rates, but not their actual values [5]. The second proposes to semantic load shedding. Reference [6] approximates the output of an operator by maximizing a user-defined similarity measure between the exact answer and the approximate answer returned by the system. Reference [18] includes QoS specifications that assign priorities to tuples and then shed those with lower priority first. Load shedding for sliding window has also been discussed in [5, 6].

6 Conclusions and Future Work

In this paper, we focus on real-time constraint of continuous query over data streams, termed as *deadline*. A *H-D-P* model is proposed, based on which *DSAT* prediction strategy is given accordingly. And a deadline-sensitive processing architecture with feedback control is proposed in which *DSAT* prediction strategy is included. Once *DIMSS* is predicted, a proposed load shedding strategy, *HopDrop*, is applied to relieve the system in advance. Experiment investigations show the performance our deadline-sensitive approach for real time processing of sliding windows. At present, our deadline-sensitive approach works in the query level, and we will try to push it into operator level. Besides, we hope to extend our deadline-sensitive approach to multi-query environment with more complicated scheduling strategy in the future work.

References

1. J. Chen, D. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continuous query system for Internet databases. In Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data, pages 379–390, May 2000.
2. R. Motwani, J. Widom, et al. Query processing, approximation, and resource management in a data stream management system. In Proc. First Biennial Conf. on Innovative Data Systems Research (CIDR), Jan. 2003.
3. S. Chandrasekaran et al. TelegraphCQ: Continuous dataflow processing for an uncertain world. In Proc. First Biennial Conf. on Innovative Data Systems Research, Jan. 2003.
4. D. J. Abadi, D. Carney, U. C. etintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, and S. Zdonik. Aurora: A New Model and Architecture for Data Stream Management. VLDB Journal, 12(2), 2003.
5. J. Kang, J. F. Naughton, and S. D. Viglas. Evaluating window joins over unbounded streams. In Proc. Int. Conf. on Data Engineering (ICDE), 2003.
6. A. Das, J. Gehrke, M. Riedewald, Approximate Join Processing Over Data Streams, In Proc. of SIGMOD 2003, June 9-12, 2003, San Diego, CA.
7. S Chandrasekaran, S Krishnamurthy, S Madden, A Deshpande, M J Franklin, J M Hellerstein, M Shah. Windows Explained, Windows Expressed.2003. Available at www.cs.berkeley.edu/~sirish/research/streaquel.pdf.
8. U Srivastava and J Widom. Flexible Time Management in Data Stream Systems. In Proc. of the 2004 ACM Symp. In Principles of Database Systems (PODS 2004), June 2004.
9. M. Hammad, W. Aref, M. Franklin, M. Mokbel, and A. Elmagarmid. Efficient Execution of Sliding Window Queries over Data Streams. Technical report, Purdue University, 2003.
10. Available at <http://ita.ee.lbl.gov/index.html>.
11. D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, and S. Zdonik. Monitoring streams—a new class of data management applications. In Proc. 28th Intl. Conf. on Very Large Data Bases, August 2002. Material augmented by personal communication.
12. J. Haritsa, M. Livny, and M. Carey. Earliest Deadline Scheduling for Real-Time Database Systems. In Procs. IEEE Real-Time Systems Symposium, pages 232-242, 1991.
13. B. Kao, H. G. Molina (1994) An overview of realtime database systems. In Stoyenko AD (ed) Real time computing. Springer, Berlin Heidelberg NewYork.
14. D. Carney, U. Cetintemel et al, Operator Scheduling in a Data Stream Systems. In Proc. of the 29th VLDB Conf., Berlin, Germany, Setp. 2003.
15. S.Babu, Shivnath, R. Motwani, K. Munagala, I. Nishizawa, J.Widom., Adaptive Ordering of Pipelined Stream Filters, In Proc. of ACM Intl. Conference on Management of Data (SIGMOD), 2004.
16. R. Avnur and J. M. Hellerstein. Eddies: Continuously Adaptive Query Processing. In Proc. of the ACM SIGMOD, Dallas, TX, May 2000, pages 261–272.
17. B. Babcock, S. Babu, M. Datar, R. Motwani, Chain: Operator Scheduling for Memory Minimization in Data Stream Systems. In Proc. of the ACM SIGMOD Int.Conf. On Management of Data, 2003: 253–264.
18. N. Tatbul, A.B. Chaudhri et al., QoS-Driven Load Shedding on Data Streams, EDBT 2002 Workshops, LNCS 2490, pp. 566–576, 2002.

MPX: A Multiversion Concurrency Control Protocol for XML Documents

Yuan Wang, Gang Chen, and Jin-xiang Dong

College of Computer Science, Zhejiang University,
Hangzhou, Zhejiang, P.R. China
wangyuanzju@yahoo.com.cn, {cg, dxj}@zju.edu.cn

Abstract. With the rapid popularity of XML, especially in web-base applications, concurrency control of XML documents becomes an important issue. In this paper, we present MPX, a multiversion concurrency control protocol for XML documents, which is designed for synchronizing concurrent queries and modifications of XML documents stored in a native XML databases or XML document repository. Distinguished from other existing protocols, MPX ensures serializability, recoverability and cascadelessness. Furthermore, MPX also guarantees read-only transactions and update transactions will never block each other, which is especially beneficial for XML for queries on XML are much more frequent than updates. Plenty of experiments have also been carried out, confirming that MPX can ensure a better degree of concurrency and less rollbacks.

1 Introduction

Due to the rapid proliferation of the eXtensible Markup Language [1], a rapidly growing number of XML documents have been produced. This is especially true in web-based applications, where the underlying data model is intrinsically semi-structured. Subsequently the need for sharing XML documents by different applications and multiple users is increasing. Concurrency control of access and update on the same XML document is thus an important issue that any XML data management system must resolve. For different XML data management system, the mechanism used for concurrency control is also different.

There are essentially three ways of storing XML documents. The simplest way is to store XML documents in a file system and access the content of them via a standardized API, such as DOM [2]. But this is not a good idea for concurrency control, for isolation must be done at the document level. The second alternative is to store them in traditional relational or object-relational database [3][4][5], where a document is shredded into one or more relational tables or stores as a CLOB or BLOB object. This seems a good idea, for concurrency control has been studied extensively in the context of traditional database management systems. Unfortunately, existing concurrency control methods used for traditional DBMS have been shown to be insufficient for XML documents [6]. The third alternative is to implement a native XML database management system. Though native XML database does not mean to be a replacement of traditional database, it's really attractive for XML data

management, since everything can be tailored to the best for manipulation of XML documents. Therefore this idea has been adopted by more and more researchers and quite a number of native XML databases has been built in last several years [7][8][9][10].

Despite of the popularity of native XML databases, little work has been done under the umbrella of concurrency control of XML documents. This is partly because that query is much more frequent than update in XML environment, and the standard query language for XML, namely XQuery [11], lacks the functionality of update. But the need for updating XML does exist, so several “unofficial” update languages for XML has been proposed [11] and W3C has also proposed its requirements for XML update language recently [12]. Therefore, synchronization and concurrency control mechanism is crucial for native XML database. Based on these observations, we introduce a multiversion concurrency control protocol, namely MPX, which can be integrated into a native XML database and provide superior concurrency control functionality.

The remainder of this paper is organized as follows. Section 2 reviews current work on XML concurrency control. Section 3 defines the data model and transaction model used by MPX. Section 4 presents the core protocol of MPX. Section 5 is experimental evaluation and its result, and section 6 wraps up the paper with a conclusion.

2 Related Work

Concurrency control is well studied for traditional database, and several effective protocols has been proposed, such as two-phase locking, timestamp-ordering and tree locking protocols etc [13]. Tree Locking protocols are often used for hierarchical data [14], but it’s not suitable for XML actually, although XML is also roughly a tree [6].

However, processing of concurrent querying and updating of XML data has received no attention for a long time, even in most native XML database management systems. For example, the Lore database management system for semistructured data and XML simply use page-based strict two phase locking for concurrency control [15][16]. Other native XML databases, such as Tamino, use locks on the document level.

Recently, with the advent of native and XML-enabled databases, several researchers picked this topic up. T. Grabs, J. Hidders and G. Moerkotte are forerunners in this area. In last 2 years, they proposed several protocols.

A group of protocols can be classified as path-based protocols, including DGLOCK [18] proposed by Grabs et al. and the transaction model proposed by J. Hidders et al [19][20]. DGLOCK is a protocol for semantic locking on DataGuide [21], which is a combination and derivation of graph-based locking protocol and predicate lock. The distinguishing characteristic of DGLOCK is that it locks path index instead of the document itself. However, DGLOCK is insufficient for handling IDREFs and the predicate lock it used is expensive and restrictive. Moreover, whether it is effective for a native XML database has not been shown. As an alternative, J. Hidders et al. proposed a path locking protocol in [19] and [20]. For simple path query without predicates, two path locking schemes, namely path lock propagation

scheme and path lock satisfiability scheme can be used. However, predicate and reference via IDREFs in path expression are not mentioned and may be troublesome. The common limitation of path-based protocols is that they are bound with a path based language and complex path expressions can hardly be treated efficiently.

Other existing protocols can be classified as node-based protocols. In this category, G. Moerkotte et al proposed several lock-based and timestamp-based protocols. In [22], three lock-based protocols, namely Node2PL, NO2PL and OO2PL, are proposed, which can ensure serializability. These protocols are intensive in use of locks, thus may bring a burden to the system. In [23], two timestamp-based protocols, namely, XTO and XCO, are proposed. All protocols proposed in [22] and [23], however, are meant to be used with a specific API (DOM). Moreover, the treatment of IDREFs is complex and in timestamp-based protocols, cascadelessness can not be guaranteed.

Despite of the limitations we have mentioned, all existing protocols are simple and easy to be integrated into a native XML database management system, which is very nice. However, we argued that all existing protocols are not well tailored for XML environment, where query is much common than update. In all existing protocols, a single update operation can block a number of queries, which prohibit the system from achieving a better degree of concurrency.

On the contrary, the protocol we proposed, MPX, solved the above problems based on multiversion. MPX is simple but powerful, it can guarantee serializability and cascadelessness. And in MPX, read-only transactions and update transactions will never block each other.

3 Preliminaries

In this section, we present some essential concepts before the core protocol of MPX is introduced.

3.1 Data Model

Many data models have been used for XML data, e.g. OEM, DOM and the XPath data model, among which the XPath data model is most suitable for native XML databases. Our data model is derived from the XPath data model, with some modifications for simplicity and extensions for multiversion. In our data model a document is a directed graph, which is defined as follows.

Definition 1. (document) A XML document d is a tuple (N, E, r) where N is the set of nodes, $E : N \times N$ denotes the set of tree edges and $r \in N$ is the root.

The data model for a document is similar to other data models, but in our data model, the content of a node is quite different, since a node can have multiple versions. Different versions of the same node can be totally different, except that they must have the same node identifier.

Definition 2. (node) A node n is a sequence of versions with the same identifier $(id, \langle n_1, n_2, \dots, n_m \rangle)$, and each version n_k is a tuple (l, v, τ, d) , where l is the label of n_k , v is the value of n_k , τ is the version number of n_k and d is a boolean

value indicating whether n_k has been deleted. For ease of illustration, we take l , v , τ , d as functions. So in the following text, we will use $l(n_k)$, $v(n_k)$, $\tau(n_k)$ and $d(n_k)$ to denote the label, value, version number and the existence of n_k respectively. $\tau(n_k)$ is added for the purpose of concurrency control, which will be discussed in next section.

For simplicity, we do not distinguish among elements, attributes and texts. For an element node n_k , $l(n_k)$ is the name of it and $v(n_k)$ is not defined. For an attribute node $l(n_k)$ is the name of it, but with a prefix “@”, and $v(n_k)$ is the value. For a text node n_k , $l(n_k)$ is defined to be “#text” and $v(n_k)$ is the literal value of it.

Our data model is extremely flexible. For example, elements can be renamed by modifying the l attribute. Elements can also be changed into texts by changing the l attribute to “#text”. The flexibility of our data model make it sufficient to match all requirements in the W3C’s official document [1].

3.2 Transaction

Definition 3. (transaction) A transaction t is a series of read or write operations, with a commit operation in the end. A read-only transaction is a transaction without write operations. An update transaction is a transaction with at least one write operations.

A read operation of node n is denoted by $r(t, n)$, similarly a write operation is denoted by $w(t, n)$. The write operation $w(t, n)$ is rich in meaning; it can be a modification, insertion or deletion. But for concurrency control they are treated roughly in the same way.

Not all sequences of operations are reasonable in practice. For example, based on XQuery and DOM, a transaction must navigate to a node step by step, starting from the root. In order to capture the characteristic of feasible sequences of operations, we define the concept of valid transaction.

Definition 4. (valid transaction) A transaction is said to be valid iff it satisfies the following requirements. 1) Before the transaction write a node, it must read it first so long as the write operation is not an insertion; 2) Before a transaction read a node, it must read all ancestors of it.

The first proposition is straightforward, for before a transaction updates or deletes a node, it must traverse to it first, and traverse is a kind of read. The second proposition is also reasonable, for conceptually XPath and XQuery queries are executed in a top down manner. That is, the execution of a XPath and XQuery queries always starts from the root of the document and searches downward in the document tree based on the path expression. This is also the truth for DOM, another popular API for XML. Though there can be IDREF and IDREFS links in the document, and the path expression can navigate through links, the usage of them is really rare in typical XML applications. And if this does occur, we can make the transaction valid simply by adding read operations of the ancestors of the link target. So in this paper, we only concentrate on valid transactions, without loss of generality. The same solution can be applied to some indexing mechanisms that ‘jump to’ nodes directly, such as DataGuide.

4 Protocol

Based on the data model presented in last section, we will discuss our concurrency control protocol for XML documents in detail here. Our protocol MPX, which stands for **M**ultiversion concurrency control **P**rotocol for **X**ML, is a combination of lock-based protocols, graph-based protocols and multiversion-based protocols in concurrency control.

MPX relies on a global version number τ for its functionality. τ is just a counter that never decrease. Each transaction t is also assigned to a version number, which is the value of the global version number τ just before its execution. The version number of t is denoted by $\tau(t)$. Each version n_k of a node n has a version number $\tau(n_k)$ too.

MPX differentiates between read-only transactions and update transactions. Suppose a transaction t issues an operation on node n , MPX operates as follows.

If t wants to read the content of n , that's to say, it issues a $r(t, n)$ operation. If t is a read-only transaction, then the content returned is the most recent version of n created by a committed transaction before t or the version created by t itself. In other words, the content read by $r(t, n)$ is n_k , which satisfies the following requirements:

1. $\tau(n_k) \leq \tau(t)$;
2. For each version $n_j, (j > k)$, $\tau(n_j) > \tau(t)$ holds;
3. Version n_k has not been deleted, that is, $d(n_k) = false$.

In some case, the required version n_k does not exist. For example, node n has just been added to the document by another transaction t' and t' is still active or has aborted. In this situation, the read operation is simply ignored.

If t is an update transaction, then t should acquire a share lock on n first. After getting the lock, it reads the latest valid version of n as in a read-only transaction.

If t issues a write operation $w(t, n)$ on node n , it should acquire an exclusive lock on n first. If the write operation is an insertion, an exclusive lock on its parent should also be acquired. If the lock conflicts with a lock of another transaction, t will be blocked until the conflict has been resolved or killed by the deadlock detector.

After t gets the lock successfully, it operates in the following way.

1. If $w(t, n)$ is an insertion, t creates the initial version of n and set its version number to ∞ ;
2. Otherwise, find the same version n_k to be update as in read operation.
3. If $\tau(n_k) = \infty$, the content of n_k is simply overwrite by t , since n_k must be the version created by t itself; otherwise, t creates a new version of n , namely n_j , and set the $\tau(n_j)$ to ∞ , a value that greater than any possible valid version number. If the write operation is a deletion, $d(n_j)$ is set to true.

At the time when an update transaction t commits, it carries out the following operations in order:

1. For each version n_k created by t , sets $\tau(n_k)$ to $\tau + 1$;
2. Increments τ by 1;
3. Releases all locks held by t .

If an update transaction t aborts, it simply releases all locks it holds. For the version number of all versions create by t is ∞ , it will be ignored base on MPX.

Since the locks acquired by an update transaction will be held until it commits or aborts, MPX follows the rule of rigorous two-phase locking for update transactions.

The protocol is not complex, but it ensures several “nice” features, which are depicted in the following theorems.

Theorem 1: If all transactions are valid transactions, MPX can ensure serializability, recoverable and cascadelessness.

Sketch of proof

For a read-only transaction t_1 and an update transaction t_2 , if t_1 starts before t_2 commits, all versions created by t_2 are “invisible” to t_1 , which is equivalent to executing all operations of t_1 first. Similarly, if t_1 starts after t_2 commits, all versions create by t_2 are “visible” to t_1 , which is equivalent to executing t_2 first. Moreover, since a read-only transaction never reads “dirty” version created by uncommitted update transaction, recoverable and cascadelessness are obviously guaranteed. For two update transactions, since MPX use the combination of two phase locking and tree-based concurrency control protocol, it’s also easy to see that serializability, recoverable and cascadelessness are ensured.

Furthermore, since read-only transactions never acquire any lock, they are never blocked by update transactions. And at the same time, since update transactions will only be blocked for locks, they will never be blocked by read-only transactions too. That is, we have the following theorem.

Theorem 2: Read-only transactions will never be blocked base on MPX. At the same time, update transactions will never be blocked by read-only transactions.

5 Evaluation

In order to test the performance of MPX, we implement a prototype system on top of DOM. Together with MPX, two other protocols presented by G. Moerkotte, namely OO2PL and XCO, are also implemented for comparison. We choose these two protocols because that they are the most effective one for lock-based protocols and timestamp-based protocol respectively [22][23].

The dataset we used for experiments are generate by the XML generator of XMark [24] with factor 0.1, roughly 10MB in size (we have also run the test on smaller and larger dataset, just to get similar results). For read-only transactions, the queries are also drew from XMark. For simplicity, we only kept XPath expressions. For update transactions, we make 50% percent of queries to updates. For ease of implementation, the update operations just mark something to be updated without modifying the content of it, so that physical recovery is not needed if a transactions rollbacks. This is not a big issue, for we concentrate on concurrency control only, except that the transaction throughput should be a little lower for protocols that have higher abortion rate.

Each transaction, despite of read-only and update, consists of roughly 5 queries or updates. After a client finished a transaction, it sleeps for a short “thinking time”, ranging from 0 to 50 milliseconds.

5.1 Varying the Number of Concurrent Transactions

This subsection shows the performance of each protocol with different number of concurrent transactions. The experiment is conducted with 70% read-only transactions and 30% update transactions. The performance is measured in four parameters, which are depicted in Fig 1.

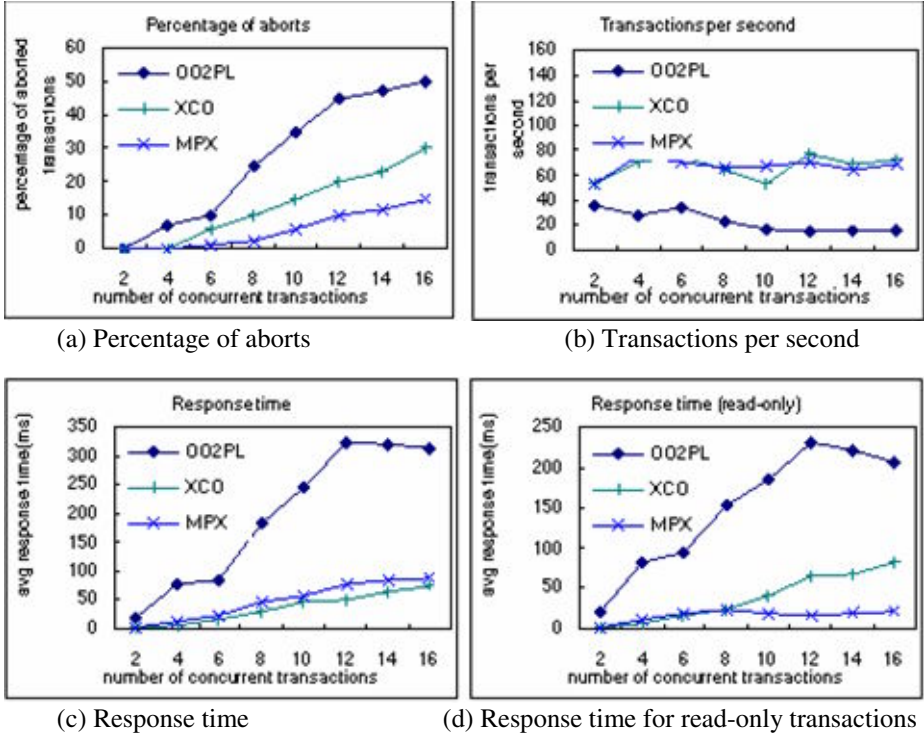


Fig. 1. Results for varying number of concurrent transactions

Fig 1(a) depicts the percentage of aborted transactions as a function of number of concurrent transactions. Obviously, more concurrent transactions will lead to higher abortion rate. But it's easy to see that in all situations, MPX is the best, followed by XCO. Similarly, Fig 1(b), Fig 1(c) and Fig 1(d) show the successful transactions per second, the average response time for successful transactions and the average response time for successful read-only transactions, respectively. As we can see in Fig 1(b), MPX and XCO tie up in the first place, while OO2PL results in a much lower transaction throughput, even in low degree of concurrency. The reason is that OO2PL will acquire much more locks than MPX, thus results in a high overhead. For response time, XCO is slightly better than MPX for it never waits for locks. However, for response time of read-only transactions, MPX is obviously the winner. And only MPX can guarantee a constant response time for read-only transactions.

5.2 Varying the Percentage of Update Transactions

Fig 2 depicts the performance of each protocol as a function of the percentage of update transactions, ranging from zero to nearly one hundred. The experiment is conducted with 10 concurrent transactions.

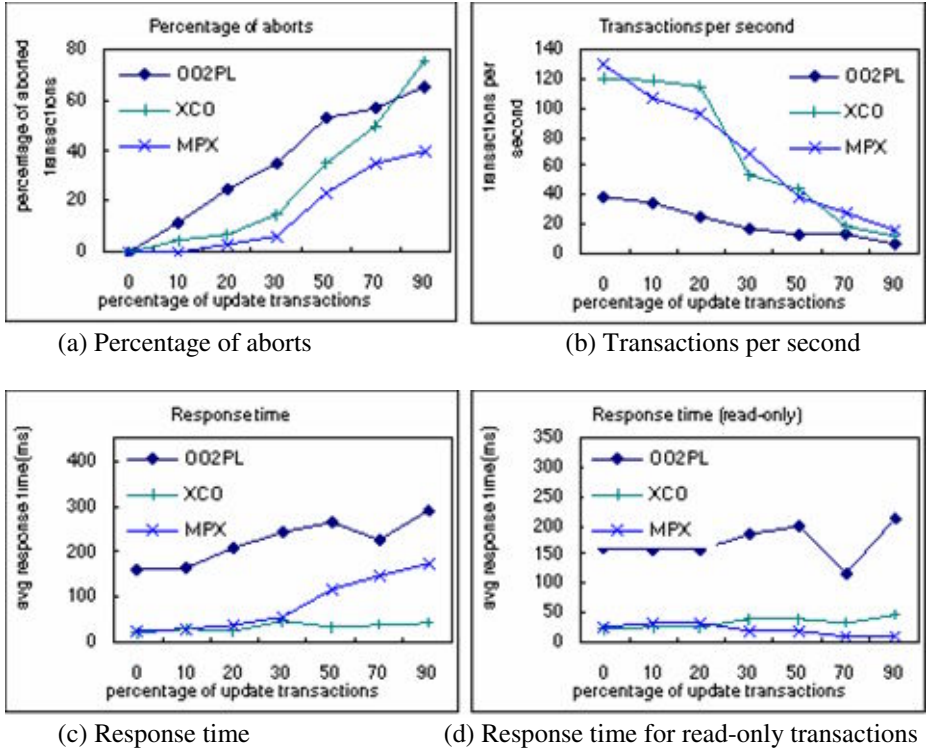


Fig. 2. Results for varying percentage of update transactions

As shown in Fig 2(a), the rate of aborted transactions grows when the percentage of update transactions become higher. In all situations, MPX leads to least abortions. XCO also performs OK when the portion of update transactions is low or median, but in high update oriented environment, it leads to the highest abortion rate. Fig 2(b) shows that the transaction throughputs of MPX and XCO are much the same, much higher than OO2PL. In Fig 2(c), we can see that when the portion of update transactions is high, OCX is the winner. This is because that conceptually MPX uses two-phase locking for update transactions and have to wait on locks. However, for read-only transactions, we can see from Fig 2(d) that MPX is the fastest. Moreover, although the response time of OCX is the lowest in high update oriented environment, its abortion rate is also very high.

5.3 Typical Query Oriented Environments

As we have stated before, most XML application are query oriented. That is, most transactions are read-only transactions. This subsection shows the performance of each protocol in those query oriented environments. The performance is also measured as a function of the percentage of update transactions, ranging only from zero to 10. Similarly, the experiment is conducted with 10 concurrent transactions.

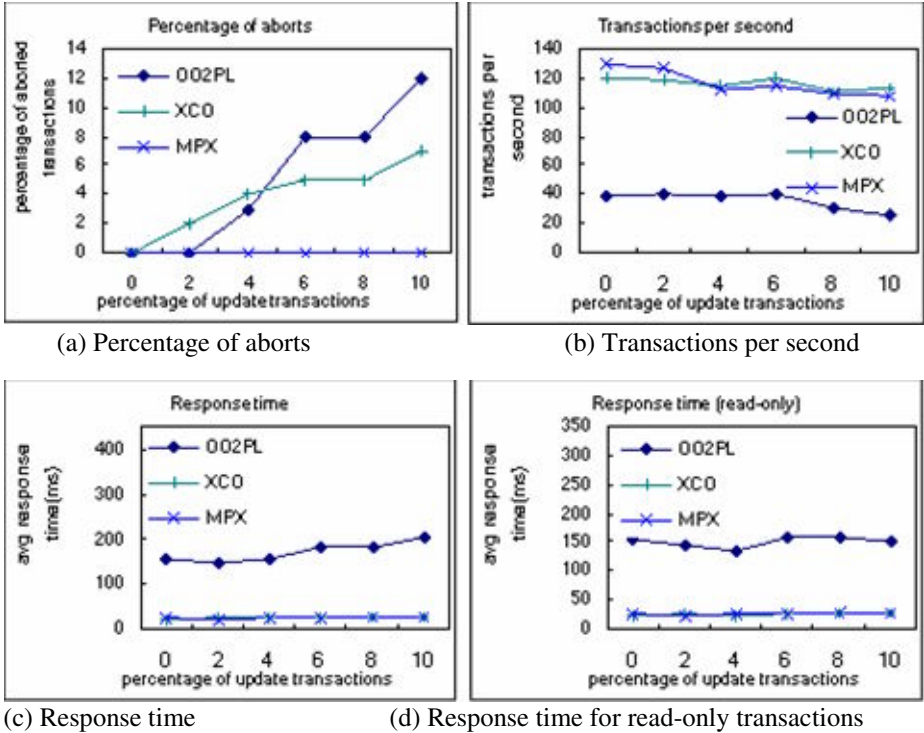


Fig. 3. Results for typical query oriented environment

As we can see from Fig 3, MPX is quite suitable for typical query oriented environments. The transaction throughput, average response time and response time for read-only transactions of MPX are much the same as XCO. However, MPX has a superior feature that its abortion rate is extremely low in query oriented environments.

6 Conclusion and Future Work

Multiversion concurrency control schemes have been used by some traditional databases (e.g. Oracle) and shown to be effective. However, its application in native XML database has not been investigated so far. In this paper, we represent a multiversion concurrency control protocol, namely MPX, for native XML database.

Except for ensuring serializability, MPX has several other nice features, such as the ability to ensure recoverability and cascadelessness. However, which distinguishes MPX from other protocols the most is that in MPX read-only transactions and update transactions never block each other.

We have conducted thorough experiments on MPX and other two existing leading protocols, OO2PL and XCO. The results show that almost in all situations, MPX wins the contest. The non-blocking of read-only transactions and update transactions contributes to less abortion, higher throughput and shorter response time eventually. And we can see multiversion shows its power in native XML database again.

However, to implement a multiversion XML database may be a tough task. For future work, we plan to convert our prototype into a real system. Therefore, more work, e.g. storage system, has to be done.

References

1. Tim Bray, Jean Paoli et al: Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, 2004
2. Arnaud Le Hors, Philippe Le Hégarret et al: Document Object Model (DOM) Level 3 Core Specification. 2004, W3C Recommendation.
3. J. Shanmugasundaram, K. Tufté, et al: Relational Databases for Querying XML Documents: Limitations and Opportunities. In Proceedings of 25th International Conference on Very Large Data Bases, Edinburgh, Scotland (1999): 302-314
4. P. Bohannon, J. Freire, et al: From XML Schema to Relations: A Cost-Based Approach to XML Storage. In Proceedings of the 18th International Conference on Data Engineering, San Jose, California, USA (2002): 64-80
5. D. Florescu, D. Kossmann: Storing and Querying XML Data using an RDMBS. IEEE Data Engineering Bulletin 22(3): 27-34 (1999)
6. Stijn Dekeyser, Jan Hidders, Jan Paredaens: A Transaction Model for XML Databases. World Wide Web 7(1): 29-57 (2004)
7. R. Goldman, J. McHugh, and J. Widom. From semistructured data to XML: Migrating the Lore data model and query language. In ACM SIGMOD Workshop on the Web and Databases (WebDB'99), Philadelphia, Pennsylvania, USA (1999): 25-30
8. H.V. Jagadish, S. Al-Khalifa, et al: TIMBER: A native XML database. The VLDB Journal, 11(4): 274-291 (2002)
9. T. Fiebig, S. Helmer, et al: Natix: A Technology Overview. Web, Web-Services, and Database Systems, Erfurt, Germany (2002): 12-33
10. H. Schöning: Tamino - A Database System Combining Text Retrieval and XML. Intelligent Search on XML Data, Applications, Languages, Models, Implementations, and Benchmarks (2003): 77-89
11. Mengchi Liu, Li Lu, Guoren Wang: A Declarative XML-RL Update Language. 22nd International Conference on Conceptual Modeling, Chicago, IL, USA (2003): 506-519
12. Don Chamberlin and Jonathan Robie: XQuery Update Facility Requirements, W3C Working Draft, 2005
13. P. A. Bernstein, V. Hadzilacos, and N. Goodman. Concurrency Control and Recovery in Database Systems. Addison-Wesley ISBN 0-201-10715-5, 1987
14. Silberschatz, A. and Z. Kedem: 1980, Consistency in Hierarchical Database Systems. Journal of the ACM 27(1), 72-80 (1980)

15. McHugh, J., S. Abiteboul et al, Lore: A Database Management System for Semistructured Data. *SIGMOD Record* 26(3), 54-66 (1997)
16. Abiteboul, S.: Querying Semi-Structured Data. In *Proceedings of the 6th International Conference on Database Theory (ICDT'97)*, Delphi, Greece (1997). 1-18.
17. H. Schöning. Tamino – a DBMS designed for XML. In *Proceedings of the 17th International Conference on Data Engineering*, Heidelberg, Germany (2001): 149–154
18. Torsten Grabs, Klemens Böhm, Hans-Jörg Schek: XMLTM: efficient transaction management for XML documents. In *Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management*, McLean, VA, USA (2002): 142-152
19. Stijn Dekeyser, Jan Hidders, Jan Paredaens: A Transaction Model for XML Databases. *World Wide Web* 7(1): 29-57 (2004)
20. Stijn Dekeyser, Jan Hidders: Conflict Scheduling of Transactions on XML Documents. In *Proceedings of the Fifteenth Australasian Database Conference on Database Technologies*, Dunedin, New Zealand (2004): 93-101
21. R. Goldman, J. Widom: DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. In *Proceedings of 23rd International Conference on Very Large Data Bases*, Athens, Greece (1997): 436-445
22. Sven Helmer, Carl-Christian Kanne, Guido Moerkotte: Timestamp-Based Protocols for Synchronizing Access on XML Documents. In *Proceedings of 15th International Conference on Database and Expert Systems Applications*, Zaragoza, Spain (2004): 591-600
23. Sven Helmer, Carl-Christian Kanne, Guido Moerkotte: Evaluating lock-based protocols for cooperation on XML documents. *SIGMOD Record* 33(1): 58-63 (2004)
24. Albrecht Schmidt, Florian Waas, et al: XMark: A Benchmark for XML Data Management. In *Proceedings of 27th International Conference on Very Large Data Bases*, Hong Kong, China (2002): 974-985

An Index Structure for Parallel Processing of Multidimensional Data

KyoungSoo Bok¹, DongMin Seo³, SeokIl Song², MyoungHo Kim¹,
and JaeSoo Yoo³

¹Department of Computer Science, Korea Advanced Institute of Science and Technology,
373-1 Guseong-dong, Yuseong-gu, Daejeon, Korea
{ksbok, mhkim}@dbserver.kaist.ac.kr

²Department of Computer Engineering, Chungju National University,
123 Iruymeon, Gumdanli, Chungju, Chungbuk, Korea
sisong@chungju.ac.kr

³Department of Computer and Communication Engineering,
Chungbuk National University, 48 Gaesin-dong, Cheongju Chungbuk, Korea
{dmseo, yjs}@chungbuk.ac.kr

Abstract. Generally, multidimensional data require a large amount of storage space. There are a few limits to store and manage those large amounts of data in single workstation. If we manage the data on parallel computing environment which is being actively researched these days, we can get highly improved performance. In this paper, we propose an efficient index structure for multidimensional data that exploits the parallel computing environment. The proposed index structure is constructed based on nP(processor)-n×mD(disk) architecture which is the hybrid type of nP-nD and 1P-nD. Its node structure increases fan-out and reduces the height of an index tree. Our proposed index structure gives a range search algorithm that maximizes I/O parallelism. The range search algorithm is applied to k-nearest neighbor queries. Through various experiments, it is shown that the proposed method outperforms other parallel index structures.

1 Introduction

In the past couple of decades, multidimensional index structures play a key role in modern database applications such as content based retrieval systems, sequence data retrievals, location based services, and so on. The applications are commonly required to manipulate multidimensional data. For example, content based retrieval systems store and retrieve multidimensional feature vector extracted by image and video. Also, location based services provide clients with the current locations of moving objects such as mobile phones. The locations of moving objects are represented as points in the two-dimensional space. To satisfy the requirements of the modern database applications, various multidimensional index structures have been proposed. There are space partitioning methods that divide the data space along predefined or predetermined lines regardless of data distributions[1, 3, 9]. On the other hand, data partitioning index structures divide the data space according to the distribution of data objects inserted or loaded into the tree[2, 4, 5, 6, 7]. Besides, Hybrid-tree[8] is a hybrid approach of data partitioning and space partitioning methods, VA-file[10] uses flat file structure, and [11] uses hashing techniques.

As mentioned above, many researchers have studied multidimensional index structures to improve retrieval performance in various ways. However, there are bounds in improving retrieval performance with a single index structure. Also, for large amount data, a single index structure may show insufficient retrieval performance. To solve these problems, several index methods using parallelism of processors or disk I/Os have been proposed [12, 13, 14, 15, 16]. These parallel multidimensional index structures can be classified into 1P-nD and nP-mD, where P and D are processor and disk, respectively. In 1P-nD architecture, multiple disks are connected to one processor so as to improve performance through parallel disk I/Os. However, there is only one channel between a processor and multiple disks so loading data to memory is processed in serial. On the other hand, in nP-mD architectures, multiple disks are connected to multiple processors. Therefore, the index structures using this architecture exploit parallelism of processors and disk I/Os.

In this paper, we propose a parallel multidimensional index structure that exploits the parallelism of the parallel computing environment. The proposed index structure is nP-n×mD architecture which is the hybrid type of nP-nD and 1P-nD. That is, there are multiple processors and each processor have multiple disks. Our node structures increase fan-out and reduce the height of an index tree. Also, range search algorithm that maximizes I/O parallelism is presented. To our knowledge, existing parallel multidimensional index structures hardly consider k-NN(k Nearest Neighbor) queries. We propose new k-NN search methods suitable to our index structures. Through various experiments, it is shown that the proposed method outperforms other parallel index structures.

The rest of this paper is organized as follows. In section 2, we describe existing parallel index structures. In section 3, we present the detailed description of our parallel high-dimensional index structure. In section 4, the results of performance evaluation are presented. Finally, we conclude in section 5.

2 Related Works

Existing parallel multidimensional index structures are classified into 1P-nD and nP-mD types. In 1P-nD architecture, multiple disks are connected to 1 processor so as to improve performance through parallel disk I/Os. MXR-tree [12] and PML-tree [14] are 1P-nD parallel index structures. These improve the performance of multidimensional index structures using the parallelism of disk I/O. MXR-tree proposed in [12] have one master server that contains all internal nodes of the parallel R-tree. The leaf level at the master does not hold the leaf level of the global tree, but (MBR, site-id, page-id) tuples for each global leaf level node. The leaf nodes of the global tree are distributed across the other servers. The (site-id, page-id) is used to locate a page and server that contains the page. Once a query is sent to the master, the master searches the internal nodes of the MR-tree and produces a list of all (site-id, page-id) pairs needed. The constructed list and the query are sent to sites containing required pages. Each site then retrieves the page from disk and sends qualifying rectangles back to the master. In [12], the requirements for improving range searches are presented. The one is *minLoad*. When the load of processing queries is light, searchers should access as few nodes as possible. Consequently, queries with small selectivity should activate as

few disks as possible. The other is *uniSpread*. Nodes that accessed by a query should be distributed over the disks as uniformly as possible. Consequently, queries with large selectivity should activate as many disks as possible. Three approaches are proposed to distribute an R-tree over multiple disks. First approach construct d independent R-trees. Second approach stripes super-node which consists of d pages on the d disks by striping pages. The last approach is MXR-tree (MultipleXed R-tree). In this approach, a single R-tree is constructed. Each node is spanned one disk page. Nodes are distributed over the d disks, with pointers across disks.

PML-tree proposed in [14] uses native space indexing with a disjoint space decomposition method. The disjoint space decomposition method does not allow overlapping intermediate MBR(Minimum Bounding Region)s. The PML-tree eliminates the extra search paths of the R-tree and the leaf node redundancy of the R+-tree by distributing data objects into multiple data spaces. Two data distribution heuristics, which distribute data over the multiple disks evenly, are proposed and implemented.

These index structures improve search performance with exploiting disk I/O parallelism. However, there is only one channel between a processor and disks so loading data to memory is processed in serial. In the nP-mD architecture, multiple disks are connected to multiple processors. nP-mD parallel index structures are constructed based on special environments such as NOW(Network of Workstation). Therefore, nP-mD index structures can use parallelism of processors and disk I/Os. MR-Tree, MCR-Tree and parallel R-tree based on DSVM(Distributed Shared Virtual Memory)[18], GPR-Tree and Parallel VA-file[17] are nP-mD parallel index structures.

MCR-tree(Master-Client R-tree) proposed in [15] reduces communication messages of MR-tree. One master and multiple clients are connected through computer network. The data structure of the master server is almost same to that of MR-tree, i.e. only internal nodes are stored at the master server. Unlike the MR-tree, only the (MBR, site-id) pairs are needed at the leaf level of the master server. Each client builds a complete R-tree for the portion of the data assigned to it. In the MCR-tree, there is redundant information stored compared to the MR-tree. This redundant information reduces the overhead at the master and global communication costs. A query is sent to the master and is processed locally as in the sequential case. When a leaf node is reached, the query MBR is sent to the client site designated in the leaf node. As soon as a client receives a request, it starts processing the query autonomously. All data are retrieved and returned back to the master. The master adds the client site id to a list that keeps track of which clients are working on the query. Since the clients work autonomously, a maximum of one request is sent to each client. The master continues searching until either all clients are notified of the query or no more MBRs intersecting the query are found. The master then waits for answers and collects the qualifying data items sent back by the clients.

3 The Proposed Index Structure

In this section, we propose an index structure for the parallel processing of multidimensional data. The proposed index structure is nP-nxD architecture which is the hybrid type of nP-nD and 1P-nD. That is, there are multiple processors and each processor have multiple disks. Figure 1 shows the architecture of the proposed parallel index structure. Disks are grouped into the number of servers evenly. The groups are

assigned to servers. One primary server coordinates search process and others are normal servers that process index operations. R-trees are distributed to servers and each server including primary server has an independent R-tree. The R-tree of each server is distributed to multiple disks. We call this architecture as $nP-n \times mD$ type. We exploit the parallelism of CPUs and each CPU uses the parallelism of multiple disk I/Os.

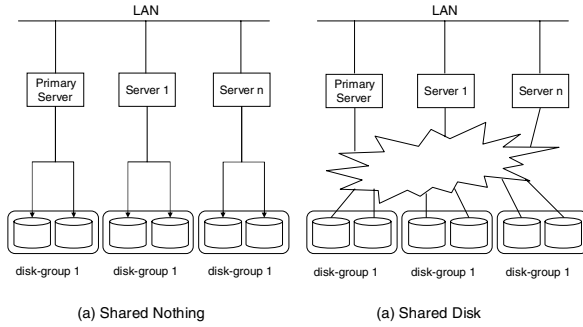


Fig. 1. Architecture of the proposed parallel index structure

Each server manages a disk group and the disk group contains an independent R-tree. Figure 2 shows the R-tree of a disk group. As shown in the figure, a node in the index structure is distributed to disks in the group, i.e., a node consists of the pages of disks. An entry in the node contains child node’s MBR and the pointers of those pages that consist of the node. In the figure, the first entry of the root node points the *node 1*. The *node 1* consists of the first pages of disk A, B and C, so the entry must have the pointers of these pages and the MBR of the *node 1*.

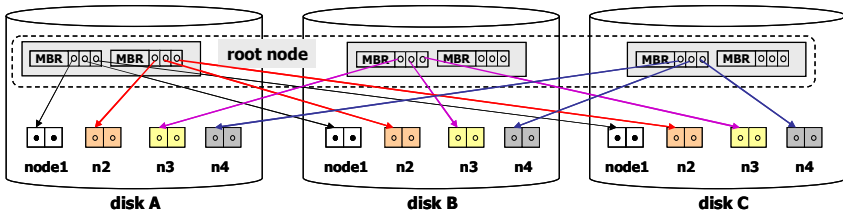


Fig. 2. Index structure of a disk group

The benefits of our architecture are as follows. First, similar data are declustered across multiple disks in the group. Since the entries in a node are distributed to multiple disks, declustering effects are maximized. Second, the height of index tree is reduced. The size of a node is determined by the page size and the number of disks in the group. As the node size increases, it takes more time to load a node into memory. However, because the index structure can load the node in parallel, the loading time is not a problem. In R-tree family, overlaps between nodes reduce the retrieval performance. The height of a tree is one of the factors to increase overlaps. As the height of a tree becomes higher, more overlaps may be caused. Finally, in multidimensional

index structures, as the dimension increases the number of nodes to be accessed increases. That is, the number of node accesses is large when processing range search or k-NN search. Subsequently, in parallel multidimensional index structure, *uniSpread* is much more important than *minLoad*. The proposed index structure read all pages that consist of a node, so it maximizes the *uniSpread*.

3.1 Insertion

Figure 3 shows the process of entry insertion. Assume that we insert entries a, b, c, d, e, f, g and h sequentially. The entries are declustered across disk-groups in round robin fashion, i.e., a is inserted into disk-group 1, b is inserted into disk-group 2 and so on. Various declustering techniques have been proposed, but in high-dimensional data sets, the performance gap among them is not so large. Also, round-robin technique is easy and cheap to implement. In that reason, we choose round-robin technique as the declustering method. Entries assigned to each group are inserted into the index structure of the group. In the first phase, we find a proper node to insert a new entry. When a node is located, we check whether the node has enough space to accommodate the entry. Then, if overflow occurs, we start split process.

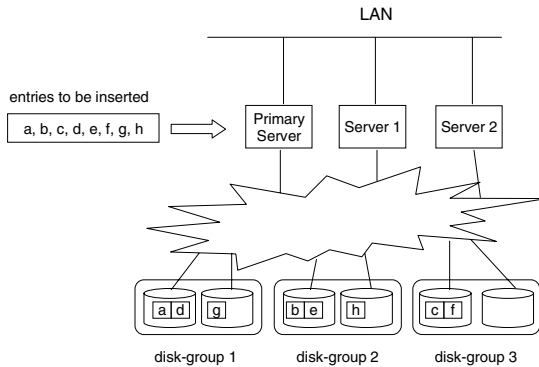


Fig. 3. Insertion of entries

When processing node split, we need to carefully allocate pages for newly created node. In general, nodes in multidimensional index structures are not always full. Consequently, we cannot fully obtain disk I/O parallelism when accessing index nodes. To relieve this problem, we place the pages of two nodes (old node and new node) in different disks as much as possible so as to increase disk I/O parallelism when processing range search. We will describe our range search algorithm in the next section. Figure 4 shows node split process. In *node 2(n2)*, overflow occurs. To split *node 2*, we assign a new node (*node 3*) and move partial entries of *node 2* to *node 3*. When allocating pages to *node 2* and *node 3*, we preferentially choose disks that have the smallest number of allocated pages. In the lower figure, disk D and E have the smallest number of allocated pages, so pages for *node 2* and *node 3* are allocated from these two disks. First, we allocate three pages from D, E and A sequentially, and then allocate three pages from B, C, and D sequentially.

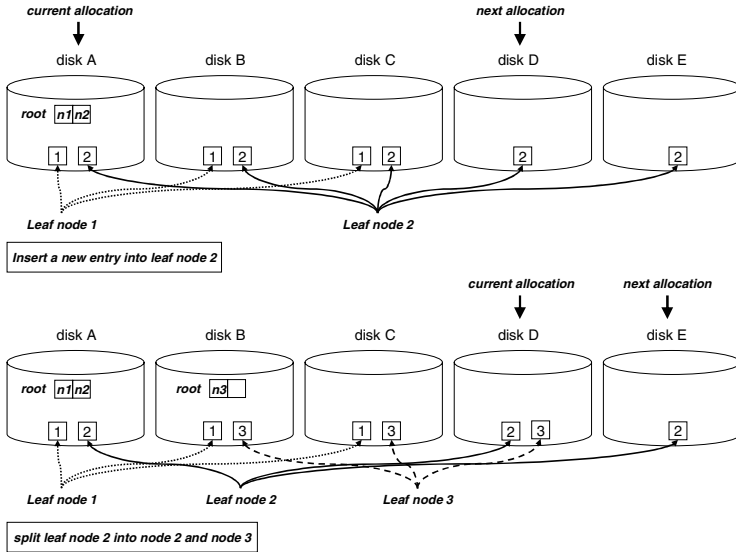


Fig. 4. Node split

3.2 Search

3.2.1 Range Search

Range search algorithms for multidimensional index structures have been mentioned in several researches [3, 6]. Searchers take multiple paths when processing range search. That is, multiple nodes may be selected as next nodes to visit. Existing range search algorithms visit the selected child nodes sequentially. Figure 5 shows the process of existing range search algorithms. A searcher chooses entries 2, 4 and 7 from root node that are overlapped with the searcher’s predicate. The searcher visit child nodes that are pointed by 2, 4 and 7 sequentially. To read node 2, the searcher must access disk A, D and E since the pages of node 2 are distributed disk A, D and E. In the similar fashion the searcher visit node 4 and 7. Total number of disk accesses is the sum of the number of disk accesses to read root node and leaf nodes. The number of disk accesses to read root node is 1 and that of leaf nodes is 3. Therefore, the total number of disk accesses to process the range query is 4.

Our new range search algorithms use different approaches to load child nodes. Once child nodes to visit are determined, we make a page loading plan according to which disks are involved to load child nodes. Figure 5 describes how to make the page loading plan. In the figure, A_3 means third page of disk A. There are 8 pages to be read. We cluster these pages into groups consists of pages from different disks. For example, pages A_3 , B_3 , C_3 , D_4 and E_1 in $GRP1$ are from different disks. It means that those pages can be read at one I/O time. Also, A_5 , D_4 and E_2 in $GRP2$ are from different disks, so we can read them in one I/O time. If we load pages in this way, only two disk I/Os are needed to load leaf nodes. One disk I/O is saved compared to the previously mentioned method.

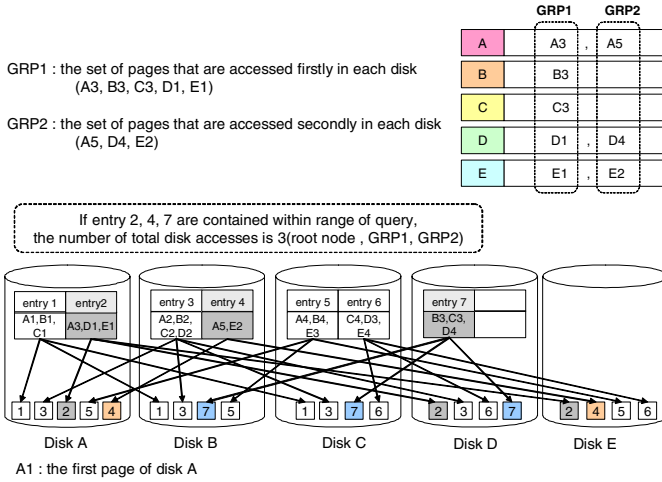


Fig. 5. Example of range search

3.2.2 k-NN Search

Existing parallel multidimensional index structures hardly consider k-NN search. However, k-NN queries are important in modern database applications. In this paper, we propose three k-NN algorithms and through experiments we show which one is the best. In the first method, the primary server distributes a k-NN query to servers and each server processes the k-NN query independently. Then, the servers return the k results to the primary server. The primary server filters the results from servers and makes final k results. The response time is the sum of the longest time among servers' response time and the time to filter servers' results. This method is simple and easy to implement. However, we may not use disk I/O parallelism like our range search algorithm because of the properties of the k-NN algorithm. When processing range search, a searcher chooses all child nodes to visit next that are overlapped with query predicates before going down to next level. Therefore, we can make a page loading plan and save disk I/Os. However, in the existing k-NN search algorithm, all child nodes to visit next are not determined definitely but just one child node is determined. Consequently, we cannot make a page loading plan as in our range search algorithm.

In the second method, the primary server transforms k-NN queries to range queries. Once a k-NN query is arrived from client, the primary server processes k-NN query partially. When the primary server gets first k results, it calculates the distance between k'th element and query point of the given k-NN query. It makes a range query with the distance. The range query is distributed to servers and the servers process the range query and return results. The primary server gathers the results from servers and makes k results. The time to process k-NN query partially is quite short. Since servers can process the transformed range query, this method can get parallelism of range search algorithm. However, the transformed range query may become larger and reduce the overall performance.

In the third method, once the primary server receives a k-NN query from clients, it sends the query to all servers. The servers execute partial k-NN queries with the re-

ceived query, transform the k-NN query to range query similar to the primary server of type 2 and return the transformed range query to the primary server. Then, the primary server redistributed the transformed query to servers. The servers process the range query and return their results to the primary server. Finally, the primary server makes k results from server's results.

4 Performance Evaluation

4.1 Experimental Setup

The simulation platform is Sun Enterprise 250 with 1GB main memory and Solaris 2.7. Simulation programs are developed with gcc 2.8 compiler. We use uniformly distributed 100,000 data with 10 ~ 80 dimensions. N_{da} means that the total number of disk accesses to perform a query. Assume that the number of disk accesses to read pages in parallel from different disks is 1. The response time to process a range query and type 1 k-NN query is calculated by the equation, $\max(RT_i) + \text{filtering time} + \text{total message size} \times T_{comm}$, where $i = 0 \sim N_{server}$, filtering time is the time to filter results from server and make final results and total message size is the size of total communication messages between the primary server and each server. The response time of type 2 and 3 k-NN queries is calculated by the following equation, $\text{query transform time} + \text{response time of a range query}$. The query transform time of type 2 is calculated by the equation, N_{da} for a partial k-NN query $\times T_{diskIO} + T_{cpu}$ for a partial k-NN query. The query transform time of type 3 is calculated by the equation, $\max(RT_i \text{ for partial k-NN query}) + \text{filtering time} + \text{total message size} \times T_{comm}$, where $i = 0 \sim N_{server}$. We assume the value of T_{comm} and T_{diskIO} as in Table 1 according to [18].

Table 1. Notations and simulation parameters

Symbol	Definition	Value
T_{comm}	communication time	1.544 Mbps
N_{disk}	number of disks	3 ~ 18
T_{diskIO}	disk I/O time to access a block	1/20,000 second
P_{size}	page size	2 ~ 48 kbyte
N_{server}	number of servers	3 ~ 15
T_{CPU}	CPU time to process a query of a server	
N_{da}	number of disk accesses	
RT	processing time for a range query of a server	$T_{cpu} + N_{da} \times T_{diskIO}$

We measure response time and total number of disk accesses of a query to compare the retrieval performance of our index structure with existing parallel multidimensional index structures. We perform several experiments in various environments. We present the results of experiments with varying dimension, the number of disks and page size. We compare our proposed index structure with MCR-tree. To our knowledge, the MCR-tree is the most recently proposed nP-mD parallel index structure and

shows best performance among existing parallel multidimensional index structures. Table 1 shows simulation parameters.

4.2 Performance Evaluation Results

We perform experiments to measure the response time and the disk accesses of k-NN queries and range queries with varying dimensions from 10 to 80, page sizes from 4k ~ 48k and disks from 3 ~ 15. Figure 6 to 8 show the response time and disk accesses of range searches and three types of k-NN searches. The graph of k-NN type 1 is omitted from the following charts since the performance gap of k-NN type 1 and others is too large to present in the charts with others. We carefully observe the performance of three k-NN queries. From the performance evaluation, we could conclude that our proposed k-NN search algorithms outperform the existing k-NN search algorithm (k-NN type 1). Also, as shown in the figures, the k-NN type 2 out performs slightly the k-NN type 1. The reason is that even though the selectivity of transformed range query in the k-NN type 3 may be smaller than that in the k-NN type 2, k-NN type 3 requires more communication messages and more CPU time to gather and filter results from the servers.

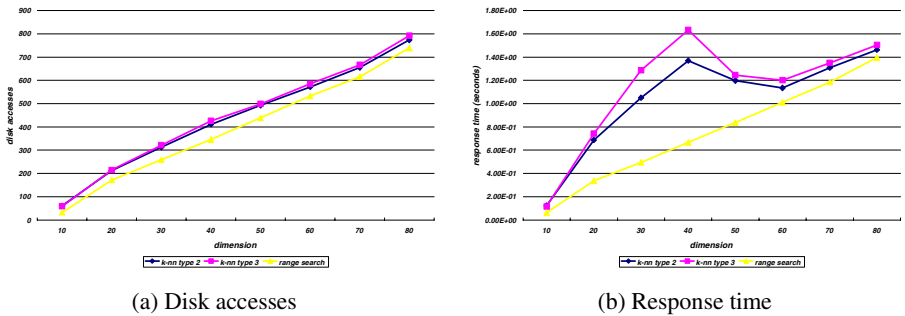


Fig. 6. Search operation with varying dimensions (data set : 100K, page size : 4k, disks : 15, servers : 3)

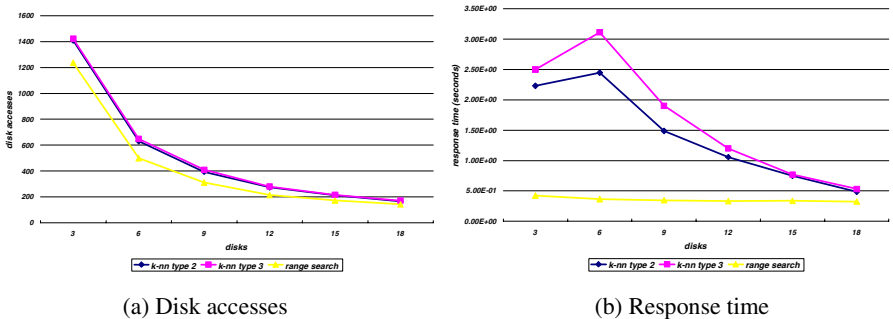


Fig. 7. Search operation with varying the number of disks (data set : 100K, page size : 4k, dimension : 20, servers : 3)

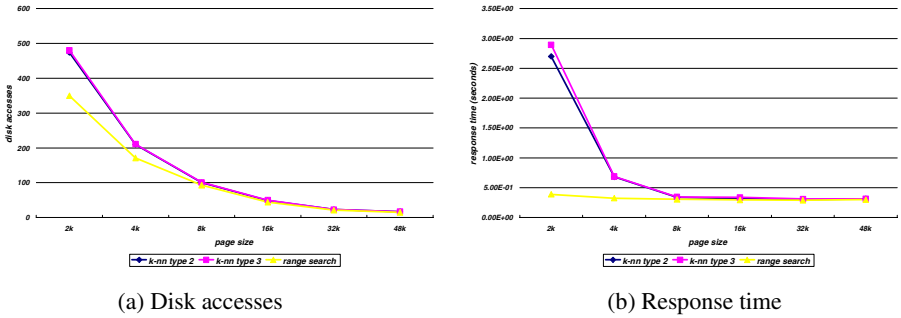


Fig. 8. Search operation with varying page size (data set : 100K, disks : 15, dimension : 20, servers : 3)

We perform various experiments to measure disk accesses and response time of the range search operations of MCR-tree and the PR-tree with varying the number of disks from 3 to 15. As shown in Figure 9, the PR-tree outperforms MCR-tree in all cases. In the MCR-tree, each server and client construct R-trees on one disk. However, we present an architecture that servers builds R-trees on multiple disks. Also, our new range search algorithms improve the disk I/O parallelism.

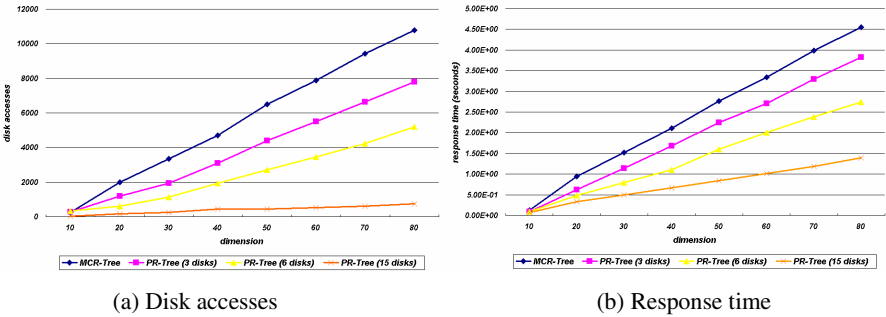


Fig. 9. Search operation with varying dimension (data set : 100K, servers : 3, page size : 4k, disks : 15)

The Table 2 shows the results of performance comparisons between k-NN search algorithms of PR-trees and MCR-trees. As shown in the figures, PR-trees outperform MCR-trees about 3 times when comparing only k-NN search algorithms. In MCR-trees, there is only one global R-tree that contains only internal nodes and leaf nodes of the global R-tree are organized as R-trees in multiple clients. k-NN search algorithms require searchers to take paths downward and backward repeatedly. Therefore, communication messages between master and clients increase. Also, since our k-NN algorithms is to transform k-NN queries to range-queries, searchers get improved disk I/O parallelism as described in the previous section.

Table 2. Disk Accesses (DA) and Response Time (RT) of search operations (dimension : 9, data set : real 100K, disks : 3, servers : 3)

Index	Range		k-NN (type 1)		k-NN (type 2)		k-NN (type 3)	
	DA	RT	DA	RT	DA	RT	DA	RT
MCR-tree	60	0.047	76	0.12	N/A	N/A	N/A	N/A
PR-tree	33	0.21	54	0.036	47	0.031	38	0.034

5 Conclusion

We proposed an efficient parallel multidimensional index structure. The proposed index structure is $nP-n \times mD$ structure that combines $1P-nD$ structure with $nP-nD$ structure. We present new range search algorithms that more efficiently use disk I/O parallelism. Even though the k-NN search are one of the important query type in multidimensional index structures, researches on improving k-NN search performance in parallel multidimensional index structures are hardly noticed. We present a new k-NN search algorithm that improves the disk I/O parallelism. Through various experiments, we prove that our proposed index structure outperforms exiting parallel multidimensional index structures.

Acknowledgement

This work was supported by the Regional Research Centers Program of the Ministry of Education & Human Resources Development in Korea.

References

1. J. T. Robinson, "The K-D-B-tree : A search structure for large multidimensional dynamic indexed", Proc. ACM SIGMOD Conference, pp.10-18, 1981.
2. A. Guttman, "R-Trees : A dynamic index structure for spatial searching", Proc. ACM SIGMOD Conference, pp.47-57, 1984.
3. J. Nievergelt, H. Hinterberger, and K. Sevcik, "The grid file : An adaptable, symmetric multikey file structure", ACM Transactions on Database Systems, Vol.8, No.1, pp.38-71, 1984.
4. N. Beckmann, H. P. Kornacker, R. Schneider and B. Seeger, "The R*-Tree : An Efficient and Robust Access Method for Points and Rectangles", Proc. ACM SIGMOD Conference, pp.322-331, 1990.
5. K. Lin, H. V. Jagadish, and C. Faloutsos, "The TV-Tree : An Index Structure for High-dimensional Data", VLDB Journal, Vol.3, No.4, pp.517-542, 1994.
6. J. S. Yoo, S. H Lee, K. H. Cho and J. S. Lee, "An Efficient Index Scheme for High-Dimensional Image Data", International Journal of Information Technology, Vol.6, No.1, pp.1-15, 2000.
7. S. Berchtold, D. A. Keim and H-P. Kriegel, "The X-tree : An Index Structure for High-Dimensional Data", Proc. VLDB, pp.28-39, 1996.
8. K. Chakrabarti and S. Mehrotra., "The Hybrid Tree : An Index Structure for High-Dimensional Feature Spaces", Proc. ICDE, pp.440-447, 1999.

9. B. Yu, R. Orlandic, T. Bailey and J. Somavaram, "KDB_{KD}-Tree : A Compact KDB-Tree Structure for Indexing Multidimensional Data", Proc. International Conference on Information Technology : Coding and Computing, pp.676-680, 2003.
10. R. Weber, H. J. Scheck and S. Blott, "Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces", Proc. VLDB, pp.194-205, 1998.
11. A. Gionis, P. Indyk and R. Motwani, "Similarity Search in High Dimensions via Hashing", Proc. VLDB, pp.518-529, 1999.
12. I. Kamel and C. Faloutsos, "Parallel R-trees", Proc. ACM SIGMOD, pp.195-204, 1992.
13. S. Berchtold, C. Bohm, B. Braunmuller, D. A. Keim and H. P. Kriegel, "Fast Parallel Similarity Search in Multimedia Databases", Proc. ACM SIGMOD, pp.1-12, 1997.
14. K. S. Bang and H. Lu, "The PML-tree: An Efficient Parallel Spatial Index Structure for Spatial Databases", Proc. ACM Annual Computer Science Conference, pp.79-88, 1996.
15. B. Schnitzer and S. T. Leutenegger, "Master-Client R-trees: A New Parallel R-tree Architecture", Proc. SSDBM, pp.68-77, 1999.
16. X. Fu, D. Wang, W. Zheng and M. Sheng, "GPR-Tree : A Global Parallel Index Structure for Multiattribute Declustering on Cluster of Workstations", Proc. APDC, pp.300-306, 1997.
17. R. Weber, "Parallel VA-File", Proc. ECDL, pp.83-92, 2000.
18. B. Wang, H. Horinokuchi, K. Kaneko and A. Makinouchi, "Parallel R-tree Search Algorithm on DSVM", Proc. DASFAA, pp.237-245, 1999.

Construction of Security Architecture of Web Services Based EAI

Di Wu, Yabo Dong, Jian Lin, and Miaoliang Zhu

College of Computer Science, Zhejiang University,
310027, Hangzhou, China
{wudi, dongyb, appolin, zhum}@zju.edu.cn

Abstract. The fast development of E-Commerce and information technology brings much higher requirements for enterprise informationization. Because of the differences in platforms, development languages and standards etc, enterprise application integration (EAI) has become the important form of enterprise informationization to support complex business processes. Web services have emerged as the next generation of integration technology which provides the power to support interoperability. However EAI doesn't just present new interoperability challenges; it also presents serious privacy and security challenges. This paper presents security architecture of Web Services based EAI which uses distributed Single Sign On authentication and distributed Role-based Access Control authorization. The basic concept and prototype system of the security architecture are described in detail.

1 Introduction

As organizations are becoming increasingly dependent on information technology, the need for integrating applications is growing. As an answer to this need, technologies in Enterprise Application Integration (EAI) have been proposed [1]. Web Services (WS), emerging as the next generation of integration technology, provides an attractive alternative for EAI. The new technology supports a loosely coupled collaboration style that eliminates many of the interoperability issues that the traditional EAI solutions have difficulty resolving [2]. The complex, heterogeneous environment of EAI doesn't just present new interoperability challenges; it also presents serious privacy and security challenges.

In the WS based EAI environment, business functions are provided as services, requests and replies of which occur between the components that are providing services instead of directly between user and end service. Suppose that a supplier accesses a business portal to buy a product order with manufacturer. However, it is the portal's service component that executes a series of transaction to each manufacturer's order management application to update the transaction/delivery status. There is a problem that requires propagation of context across multiple trust domains that are not under the control of a single enterprise.

Every service provider, in EAI, has its own security mechanism. On the one hand multi-password is a frequent consequence of multi-service environments. It often leads to violations of security policies and thus weakens the security. On the other

hand a user may be authorized to use a particular service and creates a business event to another service which may be in different geographical location.

With the problem mentioned above, this paper proposes a new approach to construct security architecture of WS based EAI, which adopts authentication based on the idea of Single Sign On (SSO) and authorization with Role-based Access Control (RBAC) model. The security architecture provides once identity recognition and explicit authorization for the same user on the different service providers of a business service chain. The standard way not only provides the necessary level of security but also simplifies the authentication and authorization at each trust layer.

In section 2, some related work about authentication and authorization are introduced briefly. The overview of WS based EAI with the security architecture is proposed in section 3. And section 4 particularly describes the key techniques to implement a prototype of the security architecture. Last a conclusion is made and prospects of the future work are looked forward.

2 Related Work

Information security management theory introduces the Confidentiality, Integrity, and Availability (CIA) concept as the guideline to implementing secure software [3]. With the requirement of Availability concept, authentication and authorization, the security problem of EAI, are mainly discussed in this paper.

In recent years, the internal security problem of enterprise applications is discussed very widely and various authentication and authorization mechanisms are fully applied. However, insufficient research has been devoted to authentication and authorization in EAI [4].

2.1 Single Sign On (SSO)

SSO is an authentication method by which users need to have an identity recognition only once and then may access authorized resources seamlessly. SSO can let users access integrated applications easier and improve their access security [5].

Today, for Web-based service environments two major approaches to SSO exist: Microsoft Passport and Liberty Alliance. The main objective of Passport [6] is the centralized storage of account information in order to simplify the login procedures and thus to ease the business activities of registered partner applications. Therefore, the Passport Server manages the authentication of users and only transmits a unique user identifier to the services. The Liberty Alliance [7] was founded to set up a SSO standard which should lead to different interoperable products from different vendors. The main objective is the coupling of multiple user identities distributed over cooperative service providers. The standard aims to support all popular operating systems, programming languages, and network structures and is designed to ensure the compatibility and security between Liberty-aware applications.

Centralized SSO approach requires all the integrated applications share the same user list. It's difficult to maintain the consistency of users involved in different integrated applications. So distributed SSO approach is adopted in our security architecture.

2.2 Role-Based Access Control (RBAC)

Role-Based Access Control (RBAC) has emerged as a proven and superior alternative to traditional discretionary and mandatory access controls [8] and is a very popular authorization mechanism in enterprise applications. As shown in Fig.1, its basic idea of authorization is that authorities are granted to roles rather than users directly and users get operation authorities from roles assigned. Because roles are more stable and have more direct understanding than users in applications, the workload and complexity of applications' administrators will be decreased greatly.

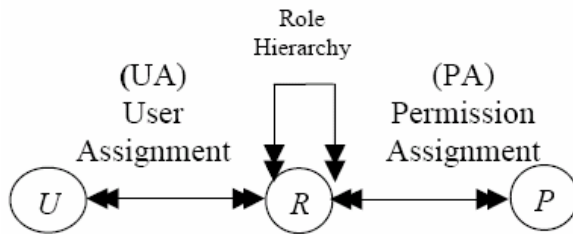


Fig. 1. Simplified RBAC

In the case of centralized systems all the authorization information is stored and managed locally in the same application where the resources reside. The most relevant problem that this scheme presents when applied to open distributed applications in EAI is the lack of interoperability. So our security architecture has built a distributed RBAC solution. Relationships between users and roles and relationships between roles and authorities are maintained by distributed applications.

3 Overview

As shown in Fig.2, EAI Virtual Environment (EVE) is a WS environment of EAI. All the integrated applications and the security architecture must be provided according to the requirements of WS. In EVE there are three kinds of services: Integrated Business Service, Integrated Security Service and Information Shared Service. Service Request (SR) can find and safely invoke business service with protection of the security architecture.

3.1 Integrated Business Service

EAI can take place on different levels, which is a fact that has been recognized since a long time. In [1], D. Linticum identifies the following four levels: data level, application interface level, method level and user interface level. The discussion in this paper falls into the scope of application interface level EAI which means an application interface is an interface that gives access to services provided by a custom application or a standard package.

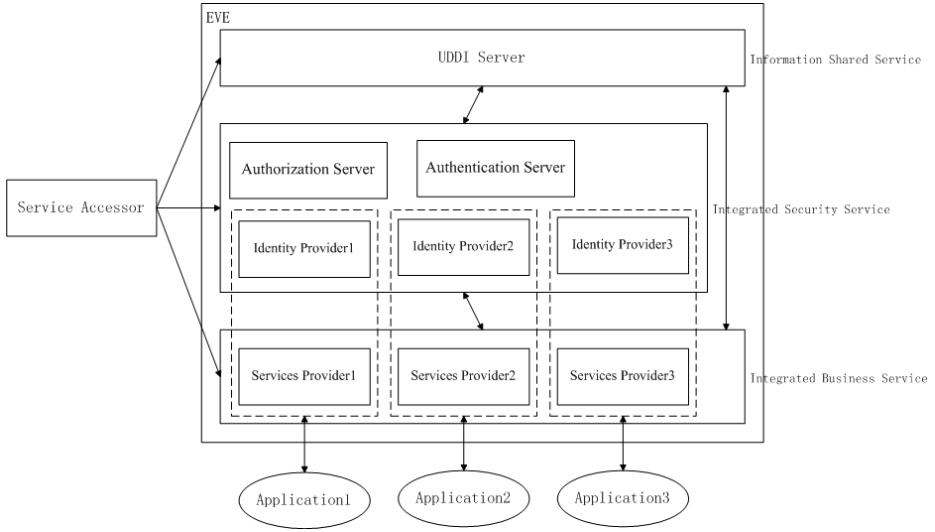


Fig. 2. WS based EAI with the security architecture

So EVE has many decentralized nodes called Service Provider (SP). All the application interfaces integrated in EVE are provided as Web Services by SP. These shared services are called Integrated Business Service (IBS).

3.2 Integrated Security Service

Services provided for authentication and authorization of access IBS are called Integrated Security Service. The security architecture of WS based EAI mainly consists of IP, AES and AOS which supports safe access with unified way in the whole EVE not only in a single application.

3.2.1 Authentication Service

EVE has two kinds of authentication node: Identity Provider (IP) and Authentication Server (AES). IP provides management of user accounts for relevant SP. Authentication is done by every IP independently in EVE. So all integrated applications must provide both integrated business services and identity service respectively. AES is responsible for distribution of authentication request to right IP and assigns an identity token to user passing verification. User can log in only once and then may access IBS with his identity token. Collaboration with AES and IP provides distributed SSO based authentication service in EVE.

3.2.2 Authorization Service

EVE has a special authorization node, Authorization Server (AOS). AOS only maintains relationships between users and roles. Before a user accesses IBS, AOS must check whether the user has access roles of the IBS. AOS provides distributed RBAC based authorization service.

3.3 Information Shared Service

UDDI Server is the centralized information shared node in EVE. Correlative information about service of AES, AOS and each SP, including the description of service classes, methods and the parameters, should be published to the UDDI Server. The shared function is called Information Shared Service. In fact, UDDI Server acts as a coordinator of other nodes.

4 Implementation and Key Techniques

According to the overview, we develop a prototype system for the security architecture and have implemented Integrated Security Service. There are several key techniques to implement the security architecture.

4.1 Distributed SSO Based Authentication Mechanism

In the security architecture, AES implements distributed SSO based authentication. Before the first authentication process begins, users need to choose the right IP where authentication will be executed. Then Service Request (SR) collects the IP information and user's identity recognition information and starts authentication process. IP information is published in AES when the application has been integrated in EVE. Then authentication can be performed normally.

The process of a successful login is depicted in Fig.3. Firstly SR sends authentication request with IP information and user's identity information (such as user name and password, etc.) to AES, secondly user's identity information will be redirected to right IP by AES, thirdly actual identity recognition is done by IP, user information will be returned to AES if user passes verification, fourthly AES saves user's information in a session and last an identity token is assigned to the user. Next time SR can access IBS with user's identity token so as to avoid authentication time after time.

Different with normal web-based SSO approach, our SSO approach for authentication in AES needs a session management itself because service requests and replies occur between the components that are providing services instead of directly between user and end service. Three kinds of session management services are required at least:

- **Time service.** The session management services related to session duration and time-out require agreement on the time.
- **User profile service.** This service provides some basic user information saved in the session.
- **Identity token issuance service.** This service grants a session ticket to an authenticated user. The identity token perhaps has some specific content and can be generated randomly too.

In session management user's identity token is the key and user's information only can be obtained with his identity token.

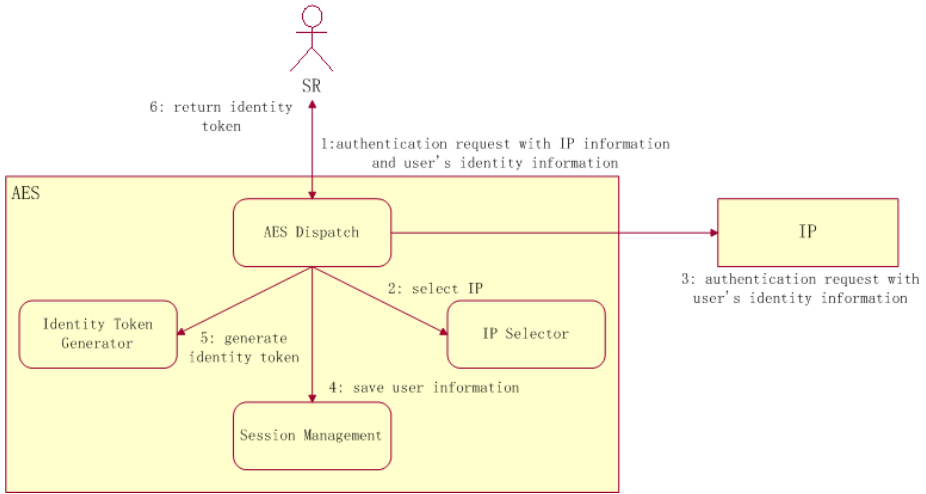


Fig. 3. Authentication working mechanism

4.2 Distributed RBAC Based Authorization Mechanism

In the security architecture, distributed RBAC based authorization has been implemented. On the basic idea of this authorization mechanism, AOS only maintains relationships between users and roles and relationships between roles and authorities are maintained by SP. Before authorization process begins, authorization records need to be set up at AOS and SP respectively. Every SP has its own authority control system that determines which resources can be accessed and how to access with the role.

4.2.1 Authorization in AOS

As shown in Fig.4, when SR accesses IBS with user's identity token, SP sends the identity token and roles binding with IBS to AOS. But AOS does not check them directly. Firstly AOS sends request to AES to get user information, secondly Session Management of AES finds user's session with the identity token, thirdly AES get user information from session and return to AOS and last Role Management of AOS verify the relationship between user and roles. SP decides user's next action on the basis of verified result.

In fact, the security architecture has two kinds of role management policy: one is a centralized role management by AOS and the other is role management by every SP itself. The latter means that SP need maintain the whole relationship among user, role and authority. SP get users' information directly from AES with their identity tokens. The centralized role management policy can integrate roles of different application that avoids repeated role definition. And the former is more flexible in management for each independent SP. It can control authority optionally and applications without RBAC support can be integrated in EVE also. Both policies are all important for the security architecture implementation and they are complementary for each other. Which one will be used is decided by SP according to actual condition.

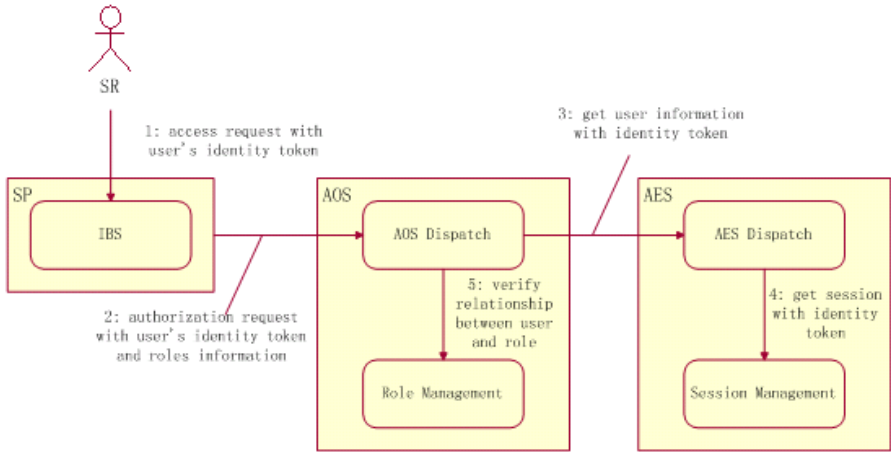


Fig. 4. Authentication in AOS

4.2.2 Authorization in SP

In the security architecture, IBS published on UDDI Server is not only the smallest unit invoked by SR but also the basic unit of authorization. SP provides authority control at the method level, which can represent various business logic of high abstract level.

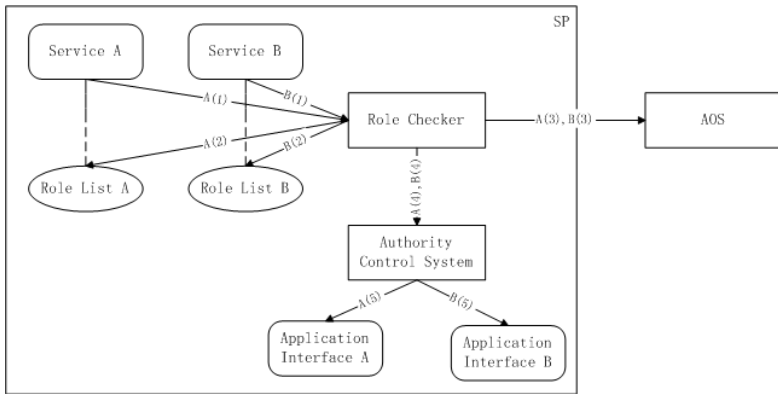


Fig. 5. Authentication in SP

The process of authorization in SP is illustrated in Fig.5. Firstly Role Checker (RC) gets the role list binding with the invoked IBS, secondly authorization process in AOS described above is executed, thirdly Authority Control System (ACS) parses the meaning of the role if the check result of relationship between user and roles from AOS is passed and last relevant application interface is enforced. Details of the ACS will not be discussed in this paper.

5 Conclusion

The security problems in open, heterogeneous and distributed applications pose important challenges. In EAI environment, a unified way is needed to simplify the authentication and authorization in the business process across multiple domains. This paper introduces security architecture of WS based EAI. It transforms the traditional centralized security mechanism to distributed one and constructs an external architecture to support security information exchange between applications.

The implementation has shown that the security architecture represents effective security guarantee in EAI environment. SSO based authentication mechanism reduces user input frequency with user name and password that decreases the possibility to lose security information because of frequent operations. RBAC based authorization mechanism protects critical resources in EAI by limiting access only to authorized users, programs, processes. It prevents the unauthorized use of a resource. In the process that user access integrated applications, only user's identity token is passed in EVE. Identity token is harder to fabricate than user information. Session Management of AES ensures that every identity token is generated by AES and fabricated ones can not work normally in EVE.

And moreover, in such security architecture, each integrated unit represents more trusting coordination. Authentication and authorization can be supported in a way that it is transparent for integrated units. Identity information exchange is performed on authentication server and actual verification is done by integrated unit independently. Each integrated unit constructs its own access control under the unified role definition on authorization server. Then the access control can be performed at diverse granularity in integrated unit. The security architecture provides a global collaboration with authentication and authorization not only in a single domain.

So WS based EAI with the security architecture makes a high request for the interaction of integrated units and meets the security needs of the EAI better.

However in heterogeneous environments where different integrated applications, using different approaches for the authorization, it could be difficult to migrate at once to a unified RBAC approach. So WS based EAI with the security architecture is more suitable for integration of new developed application than legacy system. Another point is that with applications integration increment, Role Management system in AOS will become more complicated. One user may own many roles. Perhaps plenty of definitions of relationships between role and user are reduplicate. The simplified RBAC is not enough.

The technique presented in this paper should be useful in configurable security mechanism for EAI. Developers can pay more attention to business process integration with the security architecture. Recommendations for future work are listed below: 1. To use hierarchy RBAC model to support role integration and reduce reduplicate definitions of relationships between role and user. 2. To develop a unified and configurable access control architecture to maintain the relationship between roles and authorities in every SP. 3. With the requirement of Confidentiality and Integrity concept, other security problems of EAI will be considered in our security architecture.

References

1. Linthicum, D.: Enterprise Application Integration. Addison-Wesley (2000)
2. Chung, J., Lin, K., Mathieu, R.: Web Services Computing: Advancing Software Interoperability. IEEE Computer Magazine, Vol. 36, Is: 10, Oct. IEEE Computer Society, (2003) 35–37
3. Varlamov, Stan: Security Strategies for EAI. eAI Journal, September (2002) 41–44
4. Gilliam, D.: WETICE 2003 Eighth Enterprise Security (ES) Workshop Report. 12th IEEE International Workshops on Enabling Technologies (WETICE 2003), Infrastructure for Collaborative Enterprises, 9-11 June 2003, Linz, Austria. IEEE Computer Society, (2003) 179–183
5. Volchkov, A.: Revisiting single sign-on: a pragmatic approach in a new context. IT Professional, Vol. 3, Is. 1, Jan.-Feb. (2001) 39–45
6. Microsoft. Microsoft .NET Passport Review Guide. March (2003)
7. Liberty Alliance. Liberty Architecture Overview Version 1.1. January (2003)
8. Sandhu, R., Bhamidipati, V., Munawar, Q.: The ARBAC97 Model for Role-based Administration of Roles. ACM Transactions on Information and System Security, Vol. 2, Is. 1, Feb. ACM Press, New York (1999) 105–135

Using Web Services and Scientific Workflow for Species Distribution Prediction Modeling¹

Jianting Zhang, Deana D. Pennington, and William K. Michener

LTERR Network Office, the University of New Mexico,
MSC 03 2020, 1 University of New Mexico,
Albuquerque, NM, 87131, USA
jzhang@lternet.edu

Abstract. Species distribution prediction modeling plays a key role in biodiversity research. We propose to publish both species distribution data and modeling components as Web services and composite them into modeling systems using the scientific workflow approach. We build a prototype system using Kepler scientific workflow system and demonstrate the feasibility of the proposed approach. This study is the first step towards building a virtual e-science laboratory for ecologists to perform distributed and cooperative research on species distribution predictions.

1 Introduction

The Convention on Biological Diversity (CBD) agreement signed at the United Nations Conference on Environment and Development (UNCED) held in Rio De Janeiro 1992 is a milestone towards conservation and sustainable use of biological diversity (biodiversity). Identifying species distribution and understanding global patterns of global biodiversity have become key issues in developing conservation strategies and policies [1]. During the past years, there is a massive development of biodiversity related information systems on the Internet [2] and the available species distribution data has been increased dramatically [3]. Meanwhile, considerable amount of species distribution models have been developed [4]. There are increasing needs for species distribution modeling ranging from basic ecological and biogeography research to routine conservation practices. On the other hand, most current software implementations of species distribution prediction models are developed by domain scientists which only accept ad-hoc data formats and run locally on a single machine.

With the emerging GRID technologies [5], particularly the maturing and widely adopting Web Services (WS) architecture [6], we envision a new approach to species distribution prediction: publishing both observed species presence/absence data and prediction models as Web services and then chain these Web services as scientific workflows. In this study, we aim at enabling distributed species distribution predictions with greater interoperability, flexibility and usability. While Web Services and workflow applications have been reported in other e-sciences (such as genomic

¹ This work is supported in part by DARPA grant # N00017-03-1-0090 and NSF grant ITR #0225665 SEEK.

researches [7]), we are not aware of existing modeling systems on species distribution predictions using Web Services.

This study is the first step towards building a virtual e-science laboratory for ecologists to perform distributed and cooperative research [8] on species distribution predictions. As part of a bigger project, the Science Environment for Ecological Knowledge (SEEK, [9]) – a NSF funded five year large Information and Technology Research (ITR) project, the prototype implemented for this study is built on top of several other components of SEEK project [10], particularly the EcoGrid (a collection of distributed data and analytic resources) and the Analysis and Modeling System (AMS) called Kepler (a scientific workflow system) [11].

The rest of this paper is arranged as follows. Section 2 introduces some background knowledge on species distribution prediction modeling using the Genetic Algorithm for Rule Set Production (GARP). Section 3 presents the architecture of proposed approach and discusses some implementation details. Section 4 demonstrates the feasibility of the proposed approach through a running example. Finally Section 5 is the summary and future work directions.

2 Species Distribution Prediction Modeling Using GARP

In this study, we use a specific species distribution prediction model called GARP (Genetic Algorithm for Rule Set Production, [12][13]) to demonstrate the feasibility for the proposed approach. GARP has proven especially successful in predicting species' potential distributions under a wide variety of situations [14] and have been integrated into several projects, such as Lifemapper [15] and SEEK [9]. GARP modeling requires a set of geospatial data of a study area that may be related to the species' distribution. These data sets are called environmental layers and are typically in raster format, such as elevation and precipitation. GARP also requires the observation data that recording the locations of presence or absence of species. There are three major components in GARP which are Pre-sampling, Rule generation (or Training) and Predicting. Pre-sampling produces training and testing sets by random sampling of observation data and associates the locations with the values of environmental layers at the locations. Rule generation is the process of training the genetic algorithm by the samples and generating rules that associate the values of environmental layers and the presence/absence of species for prediction. Predicting is to apply the generated rules to the values of environmental layers of all the locations of the study area and predict whether a species will be present or absent in the locations. From data mining perspective, GARP modeling can be thought as a classification or an association problem.

Traditionally, GARP systems can be run locally as a desktop application [15] or remotely through a Web browser [13]. However, all the data sets of environmental layers and all the three components of a GARP system need to be resident on a same machine. The communications between the components are often ad-hoc as well. At the same time, the volumes of environmental layer data from Geographical Information Systems (GIS) and Remote Sensing (RS) are exploding. This makes installing and running a GARP system locally inefficient since all the data need to be downloaded to the user's local machine. On the other hand, while hosting a GARP

system in a Web server avoids heavy data communication problems, it can suffer from single point failure. If any of the components in a GARP system fails working properly, it will not response to user requests any further.

In this study, we propose a scientific workflow approach to species distribution prediction modeling in distributed and heterogeneous computation environments. A scientific workflow can be seen as a scientific data analysis pipeline that connects multiple analytical steps. Compared to business workflows, scientific workflows can be data-intensive, compute-intensive, analysis-intensive, visualization-intensive, etc. There are two issues in using a scientific workflow approach to species distribution prediction modeling: The first one is that we need a workflow composition and execution environment. The second is how to represent a single analytical step in a pipeline. For the first issue, we use Kepler scientific workflow system [11][16]. For the second issue, we propose to use Web Services technology. In the proposed approach, each analytical step is implemented as a Web service and Web services are chained together to form a modeling task. In the core of Web Service technology is the Web Services Description Language (WSDL, [17]). WSDL provides a framework for defining interfaces (operations and inputs/outputs), access specification (typically Simple Object Access Protocol –SOAP [18] is used) and the endpoint (the location of the service). WSDL is written in XML and can be understood by both machine and human and can achieve greater interoperability. More specifically, we decompose the three components of GARP into Web services and multiple copies of these services can be deployed in a distributed and heterogeneous computing environment (we currently aim at supporting Windows and Linux). By using Kepler scientific workflow system, scientists can choose appropriate Web services and chain them together to compose species distribution prediction workflows and execute them either in a batch mode or interactive model as described next.

3 Architecture and Implementations

Kepler builds upon the mature, dataflow-oriented Ptolemy II system [19] which is used for modeling, simulation and design of concurrent, real-time, embedded systems. Ptolemy controls the execution of a workflow via so-called directors that represent models of concurrent computation. Individual workflow steps are implemented as reusable actors that can represent data sources, sinks, data transformers, analytical steps, or arbitrary computational steps. An actor can have multiple input and output ports, through which streams of data tokens flow. Additionally, actors may have parameters to define specific behavior. Ptolemy can perform both design-time (static) and runtime (dynamic) type checking on the workflow and data. Kepler inherits and extends these advanced features from Ptolemy and adds several new features for scientific workflows, such as prototyping workflows, distributed execution of Web and Grid services, database access and querying and supporting foreign language interfaces. We refer readers to [11][16] for more detailed information regarding to Kepler scientific workflow system.

The Web Services actor developed in Kepler serves as a proxy between the workflow system and the Web Service endpoints. There are two steps to use a Web Service actor in Kepler. The first step is to specify the URL of the WSDL of a Web service. The Web

Service actor will parse the WSDL document and retrieve available methods and their input/output types declared in the WSDL document. In the second step, users can select a method from a dropdown list. After the selection, the actor will add the corresponding ports to itself and is ready to connect its ports to ports in the other actors. We refer reader to [20] for more information of the Web Services actor in Kepler.

When a workflow is executed, the actors in the workflow will be scheduled according to the computation model of the workflow. While Ptolemy/Kepler supports a variety of computation models, two of them are frequently adopted in workflows using Web services actors, namely the Synchronous Data Flow (SDF) and the Process Network (PN). Actors need to be executed sequentially in SDF while they can be executed in parallel in PN.

The architecture of the prototype implementing the proposed approach is shown in Fig. 1. We decompose DesktopGARP [15] functions into three Web services that represent the three components in a GARP system. A database Web Services actor connecting to EcoGrid [10] which allows retrieving species occurrence data from multiple sources (museums, research institutes, etc.) is also developed. All data tokens (such as samples and rules) are encoded in XML. String data type is used for the encoded XML tokens when they pass through Kepler actors and the Web service endpoints to archive maximum system compatibility and user interpretability. The prototype was targeted at running under Apache Tomcat and Apache Axis Java. Apache Tomcat, Axis Java, Ptolemy II and Kepler are all open source based on Java. Since DesktopGARP was written in C++, we use JNI technology to wrap their native APIs into Java classes before deploying them in Apache Axis Java.

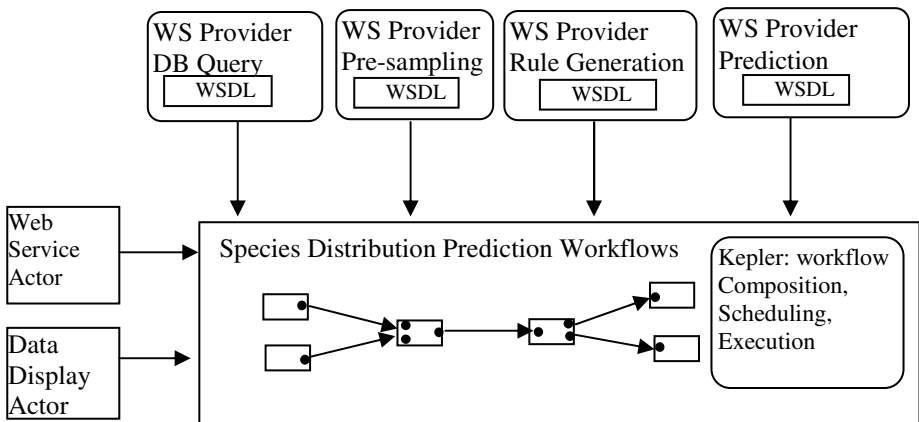


Fig. 1. Architecture

The benefits of using Web Services and scientific workflow technologies for species distribution prediction modeling can be briefly summarized as follows:

1. Efficiency. Unlike running desktop GARP systems locally, the environmental layer data management is shifted to the server side which allows using powerful database systems (such as Oracle and its Spatial option) for geospatial spatial indexing and query optimization. The communications between clients and

servers are just the pre-sampling results and the resulting rules which are generally only a small fraction of the environmental layer data sets.

2. **Interoperability.** Compared with GARP systems that adopt the browser/server architecture which targets at end users, the proposed Web Services approach is more component-oriented which allows integrating the Web services into a variety of end systems for different purposes. The Web Services approach can be treated as an extension of traditional Web approach with higher interoperability through a set of self-explained and machine-understandable WSDL documents.
3. **Robustness.** The Web Services approach allows deploying multiple copies of the GARP functional components in a distributed computation environments. If one component of a composed workflow fails working properly, it can be replaced by other similar components either manually or automatically and the workflow can still work properly.
4. **More user controls.** Kepler allows user to drag-and-drop workflow components and compose scientific workflows interactively. Users can add various display actors to view intermediate results and apply filtering and transformation actors to change the inputs to the next actor in a workflow. It also allows users to change the parameters of actors interactively and watch the changes of results. Finally, users can run workflows in a step-by-step mode to understand the executions of workflows better. Kepler scientific workflow system essentially provides a visual programming environment for species distribution prediction modeling without requiring any programming.

4 Demonstration

We use species *Mephitis* to demonstrate the proposed approach. 60 locations of observed occurrences of the species are retrieved from EcoGrid. Three instances of the Web Service actor are materialized with Pre-sampling and Prediction Web Service instances locate on one machine and Training Web Service instance locates on another machine. Once the workflow is constructed, users can execute it in Kepler scientific workflow environment by hitting the red triangle icon located on the top of Kepler window. Kepler allows users to execute a workflow in batch mode or interactive step-by-step mode. At any time during the execution of a workflow, users can hit stop and resume the execution and watch the intermediate results. Users can view the predicted distribution map using any Internet browsers. A screen snapshot is shown in Fig. 2.

A unique feather of Kepler scientific workflow is that it allows animating the execution process of a workflow which provides a user a vivid impression of how a workflow is executed (as shown in Fig. 3 where the Web service that is being invoked is highlighted). We believe this feature is important to scientists to better understand the workflows composed by their remote colleagues. Kepler is also able to output the execution schedule when users choose to open "Listen to Director" window. The execution log and any debug information will be output to the window. Each actor in a workflow has two statuses: will be iterated and was iterated. Scientists are thus able to monitor the progress of the workflow execution process. The execution schedule and execution log for the demonstration is shown in Fig. 4.

enable semi-automated compositions of scientific workflows. Finally, we believe visualization tools are important for scientist users to calibrate model parameters, interpret and evaluate prediction results and we thus plan to provide such tools in our prototype in the form of Kepler actors.

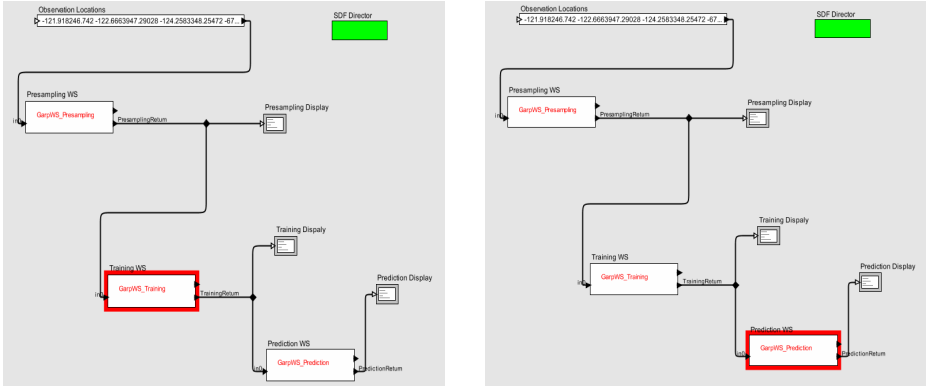


Fig. 3. Animation of Workflow Execution in Kepler for the Demonstration

```

Execute Schedule {
  Fire Actor org.sdm.spa.StringConst { .WAIM01.Observation Locations }
  Fire Actor org.sdm.spa.WebService { .WAIM01.Presampling WS }
  Fire Actor ptolemy.actor.lib.gui.Display { .WAIM01.Presampling Display }
  Fire Actor org.sdm.spa.WebService { .WAIM01.Training WS }
  Fire Actor ptolemy.actor.lib.gui.Display { .WAIM01.Training Display }
  Fire Actor org.sdm.spa.WebService { .WAIM01.Prediction WS }
  Fire Actor ptolemy.actor.lib.gui.Display { .WAIM01.Prediction Display }
}
    
```

The actor .WAIM01.Observation Locations will be iterated.
 The actor .WAIM01.Observation Locations was iterated.
 The actor .WAIM01.Presampling WS will be iterated.
 The actor .WAIM01.Presampling WS was iterated.
 The actor .WAIM01.Presampling Display will be iterated.
 The actor .WAIM01.Presampling Display was iterated.
 The actor .WAIM01.Training WS will be iterated.
 The actor .WAIM01.Training WS was iterated.
 The actor .WAIM01.Training Display will be iterated.
 The actor .WAIM01.Training Display was iterated.
 The actor .WAIM01.Prediction WS will be iterated.
 The actor .WAIM01.Prediction WS was iterated.

Fig. 4. Workflow Execution Schedule and Execution Log for the Demonstration

References

- [1] Secretariat of the Convention on Biological Diversity, Handbook of the Convention on Biological Diversity, Earthscan, 2001.
- [2] F.A.Bisby, The quiet revolution: biodiversity informatics and the Internet, *Science*, 289(5488):2309-12, 2000.
- [3] A. T.Peterson, , D. A.Vieglais, A.G.Navarro-Sigüenza, M. Silva, A global distributed biodiversity information network: Building the world museum. *Bulletin of the British Ornithologists' Club*, 123A:186-196, 2003.
- [4] A.Guisan, N.E.Zimmermann, Predictive habitat distribution models in ecology. *Ecological Modelling*. 135:147-186, 2000
- [5] Global Grid Forum, <http://www.gridforum.org/>
- [6] Web Service (WS), <http://www.w3.org/2002/ws/>
- [7] M.Claudia Cavalcanti, etc., Managing structural genomic workflows using Web services, *Data & Knowledge Engineering*, 53:45-74, 2005
- [8] W.E Johnston, Semantic services for grid-based, large-scale science, *IEEE Intelligent Systems*, 19(1):34 – 39, 2004
- [9] The Science Environment for Ecological Knowledge (SEEK), <http://seek.ecoinformatics.org/>
- [10] S.Romanello, etc., Creating and Providing Data Management Services for the Biological and Ecological Sciences: Science Environment for Ecological Knowledge, to appear in the 17th International Scientific and Statistic Database Management (SSDBM) Conference, Santa Barbara, California, USA, June 27-29, 2005
- [11] Kepler Scientific Workflow System, <http://www.kepler-project.org/>
- [12] D.R.B.Stockwell, I.R.Noble, Induction of sets of rules from animal distribution data - a robust and informative method of data-analysis, *Mathematics and Computers in Simulation* 33 (5-6): 385-390, 1992
- [13] D.R.B.Stockwell, D. Peters, The GARP Modeling System: problems and solutions to automated spatial prediction. *International Journal of Geographical Information Science* 13(2):143-158, 1999
- [14] R. P.Anderson, D. Lew, A. T. Peterson, Evaluating predictive models of species' distributions: criteria for selecting optimal models. *Ecological Modelling*, 162:211-232, 2003
- [15] Lifemapper, <http://www.lifemapper.org/>
- [16] I.Altintas, C.Berkley, E.Jaeger, M.Jones, B.Ludäscher, S.Mock, Kepler: An Extensible System for Design and Execution of Scientific Workflows, the 16th International Scientific and Statistic Database Management (SSDBM) Conference, 423-424, 2004
- [17] Web Services Description Language (WSDL), <http://www.w3.org/TR/wsdl>
- [18] Simple Object Access Protocol (SOAP), <http://www.w3.org/TR/soap/>
- [19] Ptolemy II, <http://ptolemy.eecs.berkeley.edu/ptolemyII/>
- [20] Ilkay Altintas, Efrat Jaeger, Kai Lin, Bertram Ludäscher, Ashraf Memon: A Web Service Composition and Deployment Framework for Scientific Workflows, the Second IEEE International Conference on Web Services (ICWS), 814-815, 2004

Integrative Security Management for Web-Based Enterprise Applications^{*}

Chen Zhao, Yang Chen, Dawei Xu, NuerMaimaiti Heilili, and Zuoquan Lin

Department of Information Science, Peking University, Beijing 100871, China
{zchen, imchy, xudw, nur, lz}@is.pku.edu.cn

Abstract. In enterprise environment, security becomes increasingly important and costly. Enterprises are struggling to protect the increasing amount of disparate resources. Simple patchwork of security controls no longer suffices. Enterprises require a comprehensive solution that provides centralized security management, from authentication, to authorization and to auditing. To this end, we present a design and implementation of an integrative security management solution for Web-based enterprise applications, WebDaemon. It provides Single Sign-On to multiple Web applications. It also provides restricted access to Web-based content, portals, and Web applications based on Role-Based Access Control (RBAC) policies. The WebDaemon can help enterprises secure all Web resources with consistency of policy management and reduced administrative costs.

1 Introduction

The Internet and World Wide Web have ushered a business revolution, which made a fundamental shift in the way business is conducted. Although many enterprises have deployed best-of-breed Web-based enterprise applications, such as ERP (Enterprise Resource Plan), CRM (Customer Relationship Management) and SCM (Supply Chain Management), stiff competition is forcing enterprises to implement collaborative business solutions that integrate internal systems. Enterprise information portal technologies [1] have emerged to integrate those applications into a cohesive whole. Unfortunately, current security structures have lagged behind the pace of innovation. As an enterprise expands its assets and increases its exposure to a variety of users — from employees and customers to competitors and attackers, a simple patchwork of security controls no longer suffices. There is great importance to provide integrative security for disparate applications when we bring them together. Enterprises require a comprehensive solution that provides centralized security management, from authentication, to authorization and to auditing.

With the increasing number of applications in use, users are bogged down by multiple logins and required to remember multiple IDs and passwords. To create an improved user experience and simplify administration, enterprises require a centralized

^{*} This work was supported partially by NSFC (grant numbers 60373002 and 60496322) and by a NKBRPC (2004CB318000).

mechanism to manage the authentication of users. The Single Sign-On (SSO) allows users access to all authorized applications on the basis of a single authentication that is performed when they initially access the network. Furthermore, it is essential for enterprises to provide stronger authentication mechanisms such as tokens, biometrics, or X.509 certificates.

Web-based applications greatly increase information availability and ease of access. The distribution and sharing of information via the Web require the definition and enforcement of access controls, to ensure that information will be accessible only to authorized entities. However, enterprise applications come in many shapes and sizes. It is hard to implement access control properly across distributed platforms. The current practice is to manage access control policy of each application independently, but that is expensive and inconsistent. There is a necessity for enterprises to enforce consistent access control policy across all applications.

Another key issue is secure auditing which provides a mechanism to track how information is accessed, created and modified. Auditing can be enforced in many places independently, such as firewalls, operating systems, Web servers, applications and other software. What enterprises seem to lack is the ability to track the actions of a user across multiple platforms and applications.

In this paper, we will present the design and implementation of the WebDaemon, an integrative security management solution for Web-based enterprise applications. This solution provides different types of authentication methods and the Single Sign-On to multiple Web applications. The WebDaemon centralizes access control management of Web resources, and helps to lower administration costs through Role-Based Access Control (RBAC) [2, 3]. In RBAC, permissions are associated with roles, and users are made members of appropriate roles, thereby acquiring the appropriate permissions. Also, the WebDaemon tracks all user activity in multiple applications, and shows security auditing records in a central console. To integrate with existing and future technical environments, the WebDaemon supports several standards such as: SAML [4], JAAS [5], LDAP and SSL.

We have successfully deployed the WebDaemon in a large-scale enterprise¹. The WebDaemon acts as a gatekeeper for dozens of Web servers and application servers. More than 8,000 users login various systems through the WebDaemon. The WebDaemon has exhibited sound and stable performances.

The rest of the paper is organized as follows. We introduce the architecture and components of the WebDaemon in section 2. In section 3, we show the details of our implementation. We evaluate the performance of our implementation in section 4. Finally, we summarize our work.

2 Architecture

In this section, we outline the WebDaemon architecture and explain the main components of the system. As shown in figure 1, the intended audiences can be employees, partners,

¹ TCL Corporation, <http://www.tcl.com>

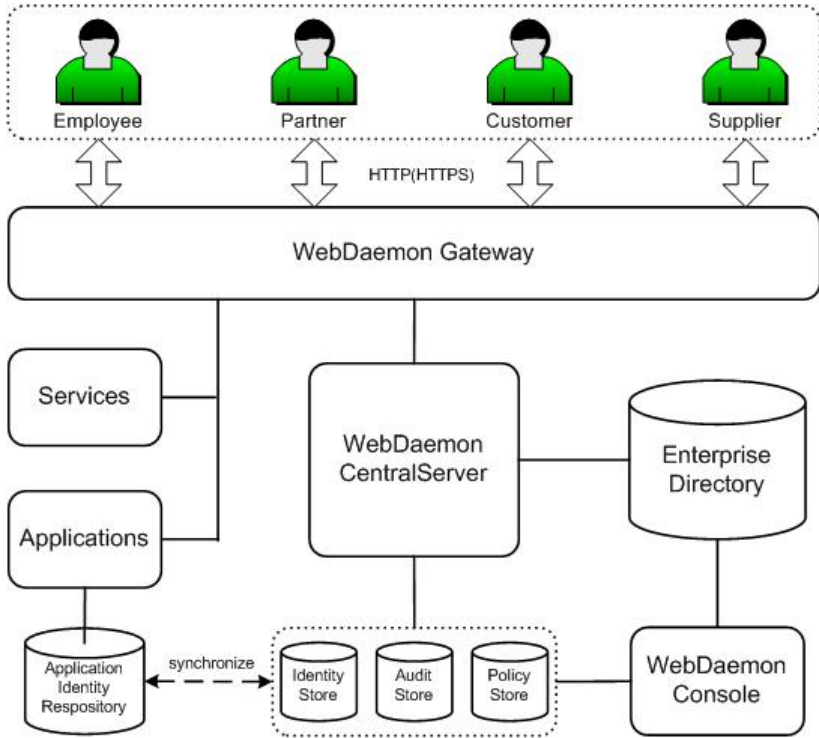


Fig. 1. Architecture of the WebDaemon

customers, or suppliers. The WebDaemon Gateway works as a reverse proxy, enforcing security controls by analyzing the HTTP protocol. The WebDaemon CentralServer provides core security services across the enterprise: authentication, authorization, and auditing. The WebDaemon Console is the service by which security administrators can configure the security policies.

To provide Single Sign-On, it is primary to aggregate and synchronize user identity information which is stored in numerous data repositories. This task can be achieved by identity management products, such as Novell Nsure Identity Manager². The result is to enable centralized administration of user identities across all enterprise applications. The directory services are used as the repository for the basic user identity information. The application-specific database maintains the repository for the information that is required only by the application. The modifications, creations and removals of identity information in master directory services are propagated to application's repository.

The WebDaemon Gateway is the unique point of Web access to disparate applications. All access requests to the resources on the backend servers are intercepted by the WebDaemon Gateway. When the WebDaemon Gateway receives a login request containing the authentication information (e.g., the hash value of user password and

² Novell Nsure Identity Manager, <http://www.novell.com/products/nsureidentitymanager/>

timestamp, or tokens), it transmits this information to the WebDaemon CentralServer. According to the response the WebDaemon CentralServer returns, the WebDaemon Gateway allows user to login or not. When an authenticated user requests a resource, the WebDaemon Gateway will send the user ID, the resource name and the request method to the WebDaemon CentralServer, which determines whether the user has the privilege to access the resource. Once the user's request is permitted, the WebDaemon Gateway will retrieve the resource from relevant backend server and send the content to the user's Web browser.

In some cases, applications require users to authenticate themselves via their own authentication modules. And it is difficult to disable or replace the authentication modules in the applications for some technical reasons. Users should provide application-specific authentication credentials to the backend server, though users have authenticated themselves to the WebDaemon Gateway, which can be regarded as primary authentication. In this situation, the WebDaemon Gateway can serve as an agent that handles the application-specific authentications automatically. These application-specific authentication credentials are stored in directory services. This mode is somewhat similar to the authentication service described in [6].

The WebDaemon CentralServer provides the decision making of security policy. It verifies whether a user's identity is authentic, whether a user can access resources and how he/she can access these resources. It also provides a centralized, cross-platform auditing service. It collects the event data from multiple applications across multiple platforms and writes the data to a single repository.

The WebDaemon supports several authentication methods: password authentication, X.509 certificate authentication and authentication token. The WebDaemon also provides extensibility for enterprises to adopt some other authentication methods. For example, Novell's NDS³ authentication service can be integrated with the WebDaemon CentralServer.

Furthermore, it is necessary to support different authentication strengths for different applications, depending on the value of the data being accessed, the location of the client or the risk profile of the user. For example, users can enter CRM system after password authentication inside the enterprise intranet, while they have to provide X.509 certificate outside the intranet. The WebDaemon provides the ability to configure all these policies.

Another key component is the WebDaemon Console. Security administrators can manage Web resources, identities, roles and privileges on it. To effectively manage the information in large enterprise, the WebDaemon Console enables security administrators to delegate certain aspects of identity and privilege administration to local administrators.

The WebDaemon Gateway locates between users and numerous Web applications. With the throughput increasing, a single gateway server will be unbearable, and the WebDaemon Gateway may become the bottleneck of the system. The WebDaemon Gateway server clusters can be configured to work in conjunction with a third-party hardware load-balancing device or load-balancing software product, such as Cisco LocalDirector⁴, to provide comprehensive load balancing and failover support.

³ Novell's NDS eDirectory, <http://www.novell.com/products/edirectory/>

⁴ Cisco LocalDirector 400 Series, <http://www.cisco.com/warp/public/cc/pd/cxsr/400/>

3 Implementation

Given the high level description of the WebDaemon architecture, we turn on describing the related technologies used in our implementation.

The WebDaemon Gateway is designed to build a highly scalable HTTP proxy server that can scale easily to thousands of concurrent clients. Considering the networking aspects such as accepting sockets, reading and writing the data from and to the network, we use the non blocking I/O facilities available in JDK 1.4⁵ to implement the WebDaemon Gateway. This allows us to save a lot of thread resources, which is not possible in traditional blocking I/O based design.

Since HTTP is a "stateless" protocol, it is difficult to differentiate between visits to a web site. Server can mark a visitor by sending a piece of state information (called a cookie [7]) to the client. Any future HTTP requests made by the client to this server should include this state. Generally, SSO in a web based intranet environment can be achieved using HTTP cookies [8]. However, there are several types of security threats to cookies. Park [9] discussed these threats in detail, and presented the design of secure cookies which can provide authentication, integrity, and confidentiality. Secure cookies are composed of a set of cookies, which contain a seal cookie that can be either MAC (Message Authentication Code) or a signed message digest of the cookies. In our implementation, the WebDaemon Gateway issues a set of secure cookies to user's Web browser after a successful user authentication. These cookies contain user's profile.

We define the WebDaemon protocol to enable the exchange of authentication and authorization information between the WebDaemon CentralServer and Gateway. The protocol is similar to SAML [4], but there are some differences between them. First, the WebDaemon protocol uses UDP as transport protocol instead of TCP, because the WebDaemon protocol is a stateless protocol as well as UDP. And UDP simplifies the server implementation. Second, the WebDaemon protocol is not based on XML, and its packet format references RADIUS protocol [10]. Thus, we need not spend time parsing XML document. Each attribute in a packet is composed of three fields "type-length-value". New attributes can be easily appended to the protocol. Third, transactions between the WebDaemon CentralServer and Gateway are authenticated through the use of a shared secret, and it is never sent over the network. In addition, all user passwords transmitted between the WebDaemon CentralServer and Gateway are encrypted. It eliminates the possibility that someone snooping on an unsecure network could determine user's password.

We also provide the SAML extension for WebDaemon CentralServer to implement Single Sign-On to external Web sites. Users can access resources outside of the enterprise without another authentication.

We use caching technique to reduce the delay experienced by the end user. The WebDaemon Gateway caches Web resources frequently requested retrieved from backend servers and authorization results returned from the WebDaemon Cen-

⁵ Java 2 SDK, New I/O, <http://java.sun.com/j2se/1.4.2/nio/index.html>

tralServer. The valid times of the cached objects can be configured. The WebDaemon Gateway can eliminate dirty records by least recently used (LRU) rule.

4 Performance

To quantify the performance impact introduced by the WebDaemon, we modeled a small Intranet environment. A 10 Mbps Ethernet connected 4 PCs, each with 2.0 GHz Pentium IV and 512 MB RAM, running Windows 2000 Server. We measured our system in two approaches.

First, we used Apache JMeter⁶ to measure the performance of the WebDaemon with caching enabled and disabled in the WebDaemon Gateway respectively. Table 1 shows the times consumed to get a static web page from a backend HTTP server. The size of the page is 6KB. In this table, the first column shows the number of threads generated by JMeter per second. The second column shows the average times consumed to get the web page directly. The third column shows access to the web page through the WebDaemon Gateway with caching enabled, and the fourth column with caching disabled. We can see that there are obvious improvements brought by caching.

Table 1. This table describes the overhead introduced by the WebDaemon with and without caching

Thread count	Access directly	Through Gateway (without caching)	Through Gateway (with caching)
1	36 ms	82 ms	40 ms
5	141 ms	372 ms	182 ms
10	306 ms	679 ms	371 ms
15	438 ms	1017 ms	521 ms

Table 2. This table describes the performance of the WebDaemon. We use LoadRunner to emulate concurrent users to login and perform 7 hits.

User count	Login time	Execution time	Response numbers/sec	CPU usage
1	0.121 s	1.704 s	33	lower than 12%
2	0.222 s	3.86 s	60	lower than 12%
5	2 s	17 s	58	lower than 16%

The other approach is using the Mercury LoadRunner⁷ to emulate large numbers of concurrent users' login and access to Web resources. By LoadRunner, we recorded the login time and the average execution time each user spent on completing 7 random hits. Table 2 summarizes the results. The first column shows the count of the

⁶ Apache JMeter, <http://jakarta.apache.org/jmeter/>

⁷ Mercury LoadRunner, <http://www.mercury.com/us/products/performance-center/loadrunner/>

concurrent users that login the WebDaemon system per second. The fourth column shows the numbers of the responses returned by the WebDaemon Gateway per second. The last column shows the CPU usage of the WebDaemon Gateway server. We can see that if there are 5 users login per second, the total number of the users logged in within five minutes will reach 1500. Each user can complete 7 hits in 17 seconds in this case. It can be seen that the performance of the WebDaemon is sufficient for large-scale enterprises environment.

In our practical environment, the backend servers comprise a DRP (Distribution Resource Planning) system, a document management system, a supply chain management system, a business reporting system, a custom service management system, and so on. Some of these systems are based on J2EE platform, some based on Microsoft IIS, and the others based on IBM Lotus Domino. We analyzed some day's log files. Figure 2 shows the numbers of the requests per hour handled by the WebDaemon in work hours.

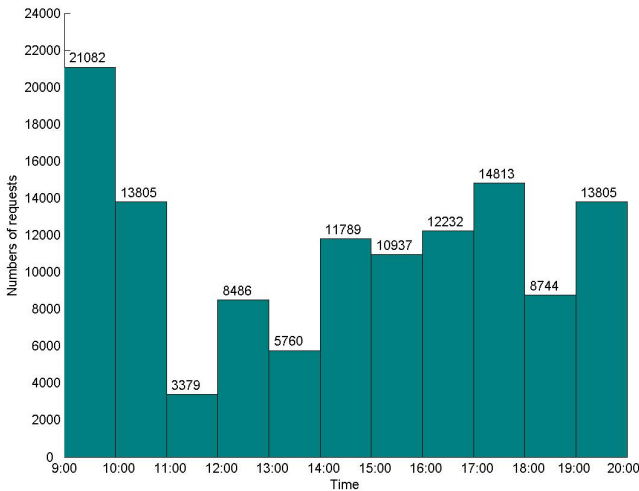


Fig. 2. Requests per hour handled by the WebDaemon in the practical environment

5 Conclusion

In this paper, we present the design and implementation of the WebDaemon, an integrative security management solution for Web-based enterprise applications. With Web Single Sign-On, Role-Based Access Control, delegated administration and centralized auditing, the WebDaemon can secure all Web resources with ease of deployment and reduced maintenance requirements. The key characteristic of the WebDaemon is the ability to centrally manage identity information and security policy. In large-scale enterprises, this capability can greatly reduce policy management inconsistencies and the cost of administration. We also show that the WebDaemon can

provide sufficient performance in large-scale enterprises environment. We have successfully deployed the WebDaemon in a large-scale enterprise and continue to support it since 2003.

References

1. Wege, C.: Portal Server Technology. *IEEE Internet Computing* **6** (2002) 73-77
2. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-Based Access Control Models. *IEEE Computer* **29** (1996) 38-47
3. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramoli, R.: Proposed NIST Standard for Role-Based Access Control. *ACM Transactions on Information and System Security (TISSEC)* **4** (2001) 224-274
4. OASIS: Security Assertion Markup Language (SAML) version 2.0. (2005) http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
5. Sun Microsystems: Java Authentication and Authorization Service (JAAS). (2003) <http://java.sun.com/products/jaas/>
6. Cox, R., Grosse, E., Pike, R., Presotto, D., Quinlan, S.: Security in Plan 9. Proceedings of the 11th USENIX Security Symposium, San Francisco (2002) 3-16
7. Kristol, D., Montulli, L.: HTTP State Management Mechanism. RFC 2965 (2000)
8. Samar, V.: Single Sign-On Using Cookies for Web Applications. Proceedings of the 8th IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Palo Alto, CA (1999) 158-163
9. Park, J.S., Sandhu, R.: Secure Cookies on the Web. *IEEE Internet Computing* **4** (2000) 36-44
10. Rigney, C., Willens, S., Rubens, A., Simpson, W.: Remote Authentication Dial in User Service (RADIUS). RFC 2865 (2000)

An Ontology-Based Semantic Integration for Digital Museums

Hong Bao¹, Hongzhe Liu¹, Jiehua Yu¹, and Hongwei Xu²

¹ Institute of Information Technology, Beijing Union University,
Beijing 100101, China
{baohong, xxtliuhongzhe}@buu.com.cn,
richard_yjh82@hotmail.com

² School of Computing, University of Paisley, Paisley, PA1 2BE, UK
hongwei.xu@paisley.ac.uk

Abstract. This paper describes the design and implementation through prototyping of the architecture of a system for browsing and retrieving museum information based on concepts. Ontology for the museum domain, based on the CIDOC Concept Reference Model, is being developed as mediation of the architecture. The challenges of developing such a global ontology model and a mapping mechanism between the global schema and data sources of local museums are discussed. Web Services technology is employed to integrate heterogeneous and distributed data sources of local museums. Experimentations with our prototype have demonstrated that this architecture is stable and efficient.

1 Introduction

Chinese civilization has been around for thousands of years, and as a result a huge amount of cultural heritage and antiques are scattered all over the vast territory of China. All kinds of digital museums have been developed for them. Each of these museums maintains large digital archives of their collections. They represent an extremely valuable cultural heritage resource, and yet access and exploitation of the data is constrained due to the distributed and heterogeneous nature of the resource. Therefore it is highly desirable to find an avenue to integrating these distributed and heterogeneous systems. Adding semantics and structures is a way of converting unstructured information into formats suitable for machine-processing. Researches from different backgrounds have attempted to provide technical support for such a conversion, and ontologies are known to be able to act as effective central mediators.

The heterogeneity of information raises several challenges for any system designed to search and retrieve information, especially if the system needs to provide such services to other organizations or the general public. The first and the most fundamental challenge is that the metadata terms used to describe and structure collections often differ from institution to institution; The second challenge is to provide access in a way that enables the digital museum visitor to exploit the richness of the data available. Previous approaches to this kind of problem are to choose a domain ontology reference model as the basis of a semantic framework, such as [1-3]. However solutions to semantic interoperability cannot be successfully implemented based solely

upon one domain ontology reference model or one theoretical framework, and there are a multitude of prerequisites, including system architectures, standards & protocols, and schema mapping rules. We learn from similar previous projects [1-3] and design our unique architecture to solve our particular problem. This project is part of a project funded by the Chinese Education Ministry which involves four simulated Local Museums in the lab of the Institute of Information Technology, Beijing Union University of China, aims to develop a system providing integrated navigation and searching capability for gallery and museum collections by adopting an ontological approach to identifying and extracting the semantics of the concepts model. The experimentations with our prototype seem to have proven that our system not only achieves anticipated goals, but also is stable and efficient.

The remainder of this paper is organized as follows: Section 2 describes the basic module of the architecture, paying particular attention to how CIDOC CRM model is used and how global ontology is constructed in our system. Section 3 gives a conclusion.

2 Architecture

We have designed the system architecture to provide integrated concept based browsing, retrieval and analysis of museum information. The architecture consists of three main parts: the Front layer provides an ontology browser; the Mediation layer which is also called the Semantic Web layer represents a global conceptual schema of antique information and maintains a mapping between the global and local schemas; and the Backend layer contains distributed Web Services and legacy databases of digital antiques. The following diagram illustrates the architecture of the system.

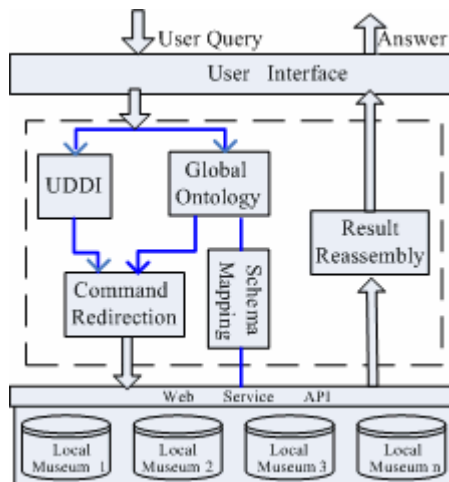


Fig. 1. System overall architecture

2.1 The Mediation Layer

Physically combining data into a single system may be impossible for technical, organizational or economic reasons. Thus mediation systems are used instead to federate information sources, which make distributed queries possible without physically aggregating information into a single monolithic database. A typical mediation system acts as a single interface for users. It accepts and interprets queries and distributes them to participating systems. These systems reply to the mediator, which then consolidates the results for the end user. In order to function correctly, a query mediation system has to be able to communicate with each participating system in such a way that it can understand and interpret the results. Participating systems are unlikely to have identical data schema and may well store different levels of detail about similar objects, so the mediation system needs to be a semantic polyglot.

The mediation layer is the key part of the system, which consists of five modules: the Global Ontology module which maintains the global ontology of our system, the Web Services UDDI module, the Schema Mapping module, the Command Redirection module and Result Reassembly module. When a user issues a query based on the Global Ontology module, the query is reformulated according to schema mapping rules recorded in the Schema Mapping module, and the reformulated query is redirected by the UDDI module which decides the related target local museums of the query according to the Web Service registration, and then executed in the distributed Local Museums. Eventually the answer is returned to the user through the Result Reassemble module.

Ontology and Reference Models for Museum Domains

Ontology is a formal shared conceptualization of domains of interest providing common vocabularies used for asserting and searching concepts [4]. A reference model is a useful representation to help the understanding, controlling and modeling of specific domains of interest. [5]. CRM (Conceptual Reference Model) [6] is one of the cultural heritage domain ontology models [7-9], and it was developed by the ICOM/CIDOC Documentation Standards Group. It describes concepts and relations relevant to the documentation of cultural domains based on an object-oriented model. It covers a vast amount of events, artifacts and people related to museums and its scope and entities are constantly modified. Using the CRM as the basis for the mediation system's data schema makes distributed query systems much easier to design. By mapping each participating system's internal data representation to the canonical form provided by the CRM, it becomes possible to integrate and interpret data stored in otherwise incompatible systems [10, 11].

Local Ontology Derivation and Global Ontology Construction

Take a Chinese antique object for example, say Bronze Gui Vessel (Fig. 2) (From Taiwan forbidden city CD-ROM published by lee & lee communications), Figure 3 shows how a local relational DB schema (left side of Fig.3) maps to the CIDOC CRM model (right side of Fig.3).



Fig. 2. Chinese antique –Bronze Gui Vessel

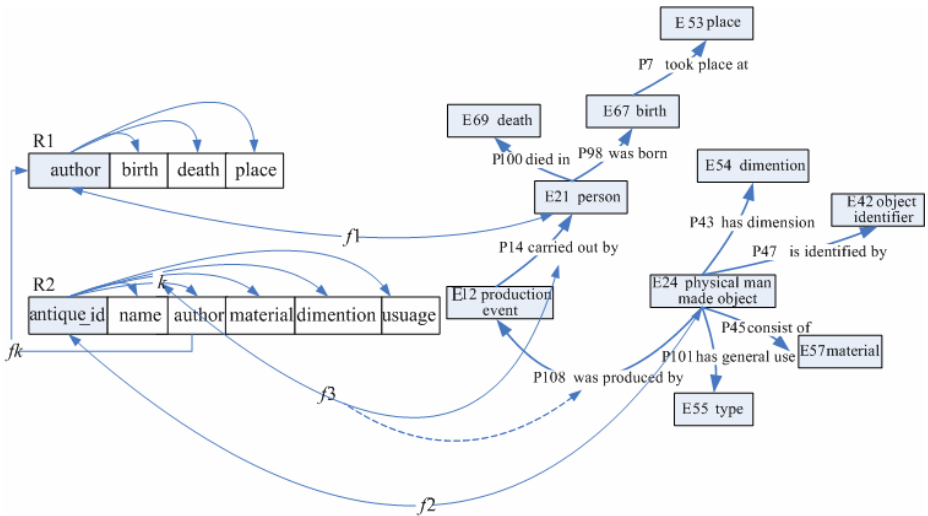


Fig. 3. Schema mapping between DB tables and CRM

The derivation process of a local ontology involves mappings between the CRM ontology and the DB schemas of local museums. That is, the semantics implicated inherently in the legacy DB logical structures and constraints can be explicitly derived by mapping them to the corresponding CRM classes and relations. These derived ontologies enrich the information content of original local schemas. For a mapping example of Figure 3, the collection of functions include a set of class mappings $f_1(E21, R1.author)$, $f_2(E24, R2.antique_id)$, and a set of derived relation mappings, e.g., $f_3((P14, P108), k)$ from the class mappings by reasoning about the CRM property declarations between $f_1.E21$ and $f_2.E24$ through an intermediate class E12. Based on the above mappings, we can get a local ontology defined in the semantics of CRM ontology.

When museums derive and develop ontologies that reflect their own viewpoints of domains, queries across the entire distributed system are still impossible. Providing a global ontology merged from the derived local ontologies (distinct copy of the CRM classes and relations) can provide a global view of the distributed system.

Schema Mapping

Given a pair of independently created schemas and asked to translate data from the source (local DB schemas) to the target (the global ontology), we consider the *schema mapping* problem which seek to interpret the correspondences in a way that is consistent with the semantics of both the source and target schemas. We achieve this in our schema mapping module which record and implement the mapping relations between the global ontology and the local museum DB sources by a mapping tool, such as the IBM Clio schema mapping tool [12]. Such a tool imports the global ontology which is represented in an XML format and the local DB schemas into its internal representations, and then accomplishes the mapping semi-automatically.

2.2 Web Services Layer

Digital resources of antiques belong to different Local Museums, which are distributed in different regions which are normally based on different computer platforms, different data formats and different programming languages. In order to share the legacy functions over the Internet to execute the reformulated query locally, we choose Web Services technology [13] to integrate these Local Museums to a single Virtual Museum. Each Local Museum acts as a service provider, and the Virtual Museum visitors act as service consumers. All the Local Museums need to do is to publish the existing functions as web service and register to the UDDI module which does the service registration and discovery. All the messages are sent using SOAP (Simple Object Access Protocol) across the entire system. The following illustrates a basic process when a visitor issues a search query from the browser:

1. The Virtual Museum Visitor issues one query on the browser.
2. The query will be passed to the UDDI module to determine which services to call and to the mediation layer to do the global ontology to local DB schema mapping, and the query is reformulated according to the mapping. After that, it is sent to command transfer module to bind and call the corresponding Web Services.
3. Each corresponding Local Museum executes called Web Services and returns their responses to the result reassembly module.
4. The query result reassembly module reassembles all the returned results from multiple Local Museums and sends them to the browser.

3 Conclusions

In this paper we have presented the architecture of a system that enables effective searching, navigating and querying the diversity of antique information held by museums and galleries. The design integrates Local Museum resources based on the CIDOC Concept Reference Model by using the Web Services technology. The archi-

ecture supports concept based approaches to querying heterogeneous museums seamlessly. Museums contain a large quantity of multimedia information including images, audio and video records, etc. In order to search the multimedia content conceptually, our next goal is to combine the main and specific aspects of MPEG-7 and CIDOC-CRM models into a single ontology for describing and managing multimedia information in museums [14].

References

1. Musinfo project: <http://www.ville-ge.ch/musinfo>.
2. RLG Cultural Material project: <http://culturalmaterials.rlg.org/cmiproduct/web/workspace.jsp>.
3. SCULPTEUR project: <http://www.sculpteurweb.org/>.
4. OntoWeb project: <http://ontoweb.ontoware.org/>.
5. Vernadat, F. B.: Enterprise Modelling and Integration, principles and applications. Chapman & Hall, INRIA Lorraine, France (1996)
6. CIDOC CRM Version 3.4.9: http://cidoc.ics.forth.gr/docs/cidoc_crm_version_3.4.9.doc.
7. Iconclass: <http://www.iconclass.nl/>
8. Getty: http://www.getty.edu/research/conducting_research/
9. ABC Ontology: http://metadata.net/harmony/JODI_Final.pdf
10. Doerr, M.: Mapping a Data Structure to the CIDOC Conceptual Reference Model. <http://cidoc.ics.forth.gr/tutorials.html>
11. Karvasonis, I.: Study of the heterogeneity in cultural databases and transformation of examples from CIMI to the CIDOC-CRM. <http://www.ics.forth.gr/publications/technical-reports>.
12. Lucian Popa: Translating Web Data. Proceeding of 2002 Very Large Databases Conference (VLDB'02), Hong Kong(2002)598-609
13. W3C Web Services: <http://www.w3.org/2002/ws/>
14. Hunter, J.: Combining the CIDOC CRM and MPEG-7 to describe multimedia in museums. Proceedings of Museums on the Web 2002 Conference, Boston (2002)

A Unified Subspace Outlier Ensemble Framework for Outlier Detection

Zengyou He, Shengchun Deng, and Xiaofei Xu

Department of Computer Science and Engineering,
Harbin Institute of Technology, China
zengyouhe@yahoo.com, {dsc, xiaofei}@hit.edu.cn

Abstract. This paper proposes a unified framework for outlier detection in high dimensional spaces from an ensemble-learning viewpoint. Moreover, to demonstrate the usefulness of our framework, we developed a very simple and fast algorithm, namely SOE1, in which only subspaces with one dimension is used for mining outliers from large categorical datasets. Experimental results demonstrate the superiority of SOE1 algorithm.

1 Introduction

Most applications for outlier mining are high dimensional domains in which the data may contain hundreds of dimensions. In this paper, we propose a unified framework for outlier detection in high dimensional spaces from an ensemble-learning viewpoint.

In our new framework, the outlying-ness of each data object is measured by fusing outlier factors in different subspaces using a combination function. In addition, to demonstrate the usefulness of the ensemble-learning based outlier detection framework, we developed a very simple and fast algorithm, namely SOE1 (Subspace Outlier Ensemble using 1-dimensional Subspaces) in which only subspaces with one dimension is used for mining outliers from large categorical datasets. The SOE1 algorithm needs only two scans over the dataset and hence is very appealing in real data mining applications. Experimental results on real datasets show that SOE1 has comparable performance with respect to those state-of-art outlier detection algorithms on identifying true outliers.

2 Problem Formulation and Unified Framework

Let D be a database of d -dimensional feature vectors. An element $P \in D$ is called point or object. Let $A = \{A_1, A_2, \dots, A_d\}$ be the set of all attributes A_i of D . Any subset $S \subseteq A$, is called a subspace. The cardinality of S ($|S|$) is called the dimensionality of S . The power set of A , denoted by $Pow(A)$, is defined as $Pow(A) = \{S \mid S \subseteq A\}$. Hence, each subspace is an element of $Pow(A)$. The projection of an object P into a subspace $S \in Pow(A)$ is denoted by $\pi_S(P)$. The outlier factor of an object P in subspace S is denoted by $OF(\pi_S(P))$.

The problem of outlier detection in high dimensional space and the unified ensemble learning based algorithmic framework are described in Fig.1. The input for outlier detection in high dimensional space includes the target database, the number of desired outliers, the set of subspaces considered in the mining process and the combination function. Among all these input parameters, the set of subspaces and combination function are of primary importance.

Outlier Detection in High Dimensional Space:

Mining top-k outliers from a database using a set of subspaces and a combination function

Input:

- (1): A database D with set of features A
- (2): An Integer k , i.e., the k most outlying objects to be mined
- (3): SS , a set of subspaces, i.e., SS is a subset of $Pow(A)$
- (4): A combination/ensemble function \oplus

Output:

Top-k outliers that satisfy the requirement

Unified Algorithmic Framework:

(1) Individual subspace outlier factor computation step

For each subspace S in SS

For each object P in D

Compute the outlier factor of P in S , i.e., $OF(\pi_S(P))$

(2) Outlier ensemble step

For each object P in D

Ensemble all the outlier factors of P in different subspaces, i.e., $OF(P) = \bigoplus_{S \in SS} OF(\pi_S(P))$

Fig. 1. The unified subspace outlier ensemble based algorithmic framework-SOE framework

The unified algorithmic framework (subspace outlier ensemble (SOE) framework) consists of two steps: subspace outlier mining and subspace outlier ensemble.

In the subspace outlier-mining step, the SOE framework uses existing outlier mining algorithms to compute the outlier factors of data objects in all the input subspaces.

In subspace outlier ensemble step, we borrow some ideas from ensemble learning by fusing outlier factors in different subspaces using a combination function. Hence, the choice of combination function (or combining operator) is at the core of the outlier ensemble stage.

Suppose the outlier factors of an object P in D in different subspaces are denoted as v_1, v_2, \dots, v_m (the number of input subspaces is m). And the combining operator is denoted as \oplus . By fusing all the subspace outlier factors, the final outlier factor of P is $OF(P) = \bigoplus(v_1, v_2, \dots, v_m)$. Note that if $m=1$, $\bigoplus(v_1, v_2, \dots, v_m) = v_1$.

Our potential choices for \oplus are the followings (which are also used in [7] for class outlier mining).

- The product operator \prod : $\bigoplus(v_1, v_2, \dots, v_m) = v_1 v_2 \dots v_m$.
- The addition operator $+$: $\bigoplus(v_1, v_2, \dots, v_m) = v_1 + v_2 + \dots + v_m$.

- A generalization of addition operator-it is called the S_q combining rule, where q is an integer number. $S_q(v_1, v_2, \dots, v_m) = (v_1^q + v_2^q + \dots + v_m^q)^{(1/q)}$. Note that the addition is simply the S_1 rule.
- A “limiting” version of S_q rules, denoted as S_∞ . $S_\infty(v_1, v_2, \dots, v_m)$ is defined to be equal to v_i , where v_i has the largest absolute value among (v_1, v_2, \dots, v_m) .

3 SOE1 Algorithm

Let D be a database of d -dimensional feature vectors. Let $A = \{A_1, A_2, \dots, A_d\}$ be the set of all attributes A_i of D . The value set V_i is set of values of A_i that are present in D . For each attribute value $v \in V_i$, the frequency $f(v)$, denoted as f_i , is number of objects $P \in D$ with $P.A_i = v$. The number of distinct attribute values of A_i is supposed to be p_i . We define the histogram of A_i as the set of pairs: $h_i = \{(v_1, f_1), (v_2, f_2), \dots, (v_{p_i}, f_{p_i})\}$. Each element of h_i is called an entry in the histogram or just a histogram entry. The histogram of the dataset D is defined as: $H = \{h_1, h_2, \dots, h_d\}$.

The proposed SOE1 algorithm needs only two scans over the dataset. The first scan of SOE1 is the subspace outlier-mining step, in which we construct the histogram of the dataset D . Intuitively, in one-dimensional space the outlying-ness of an object is determined by the occurrences of its corresponding attribute value, i.e., higher frequency implies more normal the object is. Hence, the outlier factor of each object $P \in D$ in subspace A_i is the frequency $f(P.A_i)$. Hence, $S_\infty(v_1, v_2, \dots, v_m)$ is modified to be equal to v_i , where v_i has the smallest absolute value among (v_1, v_2, \dots, v_m) . To store the histogram of the dataset D , we need d hash tables as our basic data structures (each hash table for one histogram of A_i). Actually, each hash table is the materialization of a histogram. Therefore, we will use histogram and hash table interchangeably in the remaining parts of the paper.

The second scan of SOE1 is the subspace outlier-ensemble step, in which we aggregate the outlier factors in different one-dimensional subspaces using a combination function. That is, for each object $P \in D$, we retrieve its frequencies of attribute values, i.e., outlier factors, from hash tables efficiently. Then, fusing these outlier factors to get final outlying-ness. To report the top- k outliers, we maintain a k -length array for this purpose.

4 Experimental Results

We used three real life datasets from UCI [5] to demonstrate the effectiveness of our algorithm against *FindFPOF* algorithm [1], *FindCBLOF* algorithm [2] and *KNN* algorithm [3].

For all the experiments, the two parameters needed by *FindCBLOF* algorithm are set to 90% and 5 separately as done in [2]. For the *KNN* algorithm [3], the results were obtained using the *5-nearest-neighbour*; For *FindFPOF* algorithm [1], the parameter *mini-support* for mining frequent patterns is fixed to 10%, and the maximal number of items in an itemset is set to 5. Since the SOE1 algorithm is parameter-free,

we don't need to set any parameters. Furthermore, we empirically study the impact of different combining operators on SOE1. That is, in the experiments, we report the results of SOE1 with different combining operators. For S_q operator, we set q to 2, 5 and 7 separately.

Since we know the true class of each object in the test datasets, we define objects in small classes as rare cases (i.e., outliers). The number of rare cases identified is utilized as the assessment basis for comparing our algorithm with other algorithms.

The first dataset used is the Lymphography dataset, which has 148 instances with 18 attributes. The data set contains a total of 4 classes. Classes 2 and 3 have the largest number of instances. The remained classes are regarded as rare class labels for they are small in size. The corresponding class distribution is illustrated in Table 1.

Table 1. Class Distribution of Lymphography Dataset

Case	Class codes	Percentage of instances
Commonly Occurring Classes	2, 3	95.9%
Rare Classes	1, 4	4.1%

Table 2 shows the results produced by different algorithms. Here, the *top ratio* is ratio of the number of records specified as *top-k* outliers to that of the records in the dataset. For example, we let SOE1 (+) algorithm find the *top 16* outliers with the top ratio of 11%. By examining these 16 points, we found that 6 of them belonged to the rare classes.

Table 2. Detected Rare Classes in Lymphography Dataset

Top Ratio (Number of Records)	Number of Rare Classes Included								
	SOE1 (Π)	SOE1 (+)	SOE1 (S_q)			SOE1 (S_∞)	Find FPOF	Find CBLOF	KNN
			q=2	q=5	q=7				
5%(7)	6	5	4	4	4	2	5	4	4
10%(15)	6	6	5	4	4	6	5	4	6
11%(16)	6	6	5	4	4	6	6	4	6
15%(22)	6	6	5	5	4	6	6	4	6
20%(30)	6	6	6	5	4	6	6	6	6

One important observation from Table 2 was that, among all the potential choices of \oplus we are considered in SOE1, the + operator and Π operator are the clear winners in this experiment. That is, SOE1 with the + operator and Π operator outperform S_q and S_∞ in all cases. This observation suggests that the + operator and Π operator will be better choices in practice for users. Consequent experiments also support similar conclusions. Moreover, with increase of q in the S_q operator, the performance of SOE1 will deteriorate.

Furthermore, in this experiment, the SOE1 algorithm with Π operator performed the best for all cases and can find all the records in rare classes when the *top ratio*

reached 5%. In contrast, for the *FindFPOF* algorithm, it achieved this goal with the *top ratio* at 10%, which is almost the twice for that of our algorithm.

The second dataset used is the Wisconsin breast cancer data set, which has 699 instances with 9 attributes. Each record is labeled as *benign* (458 or 65.5%) or *malignant* (241 or 34.5%). We follow the experimental technique of Harkins, et al. [6] by removing some of the *malignant* records to form a very unbalanced distribution; the resultant dataset had 39 (8%) *malignant* records and 444 (92%) *benign* records (<http://research.cmis.csiro.au/rohanb/outliers/breast-cancer/>). The corresponding class distribution is illustrated in Table 3.

Table 3. Class Distribution of Wisconsin Breast Cancer Dataset

Case	Class codes	Percentage of instances
Commonly Occurring Classes	1	92%
Rare Classes	2	8%

Table 4. Detected Malignant Records in Wisconsin Breast Cancer Dataset

Top Ratio (Number of Records)	Number of Rare Classes Included									
	SOE1 with different operators						<i>Find FPOF</i>	<i>Find CBLOF</i>	<i>RNN</i>	<i>KNN</i>
	\prod	+	(S_q)			S_∞				
			q=2	q=5	q=7					
1%(4)	4	4	3	3	3	3	4	3	4	
2%(8)	7	7	7	7	7	5	7	7	6	8
4%(16)	15	14	14	14	14	11	14	14	11	16
6%(24)	22	21	19	19	16	17	21	21	18	20
8%(32)	27	28	26	25	23	23	28	27	25	27
10%(40)	33	32	31	30	28	28	31	32	30	32
12%(48)	36	36	34	33	33	33	35	35	35	37
14%(56)	39	39	38	37	37	37	39	38	36	39
16%(64)	39	39	39	38	38	38	39	39	36	39
18%(72)	39	39	39	39	39	38	39	39	38	39
20%(80)	39	39	39	39	39	39	39	39	38	39
25%(100)	39	39	39	39	39	39	39	39	38	39
28%(112)	39	39	39	39	39	39	39	39	39	39

For this dataset, we also consider the *RNN* algorithm [6]. The results of *RNN* algorithm on this dataset are reproduced from [6]. Table 4 shows the results produced by the different algorithms. Clearly, SOE1 with + operator and \prod operator also outperform SOE1 with S_q and S_∞ in all cases on this dataset. Furthermore, among all of these algorithms, *RNN* performed the worst in most cases. Compared to other algorithms, SOE1 (with + operator and \prod operator) achieves roughly the same average performance with respect to the number of outliers identified.

Arrhythmia data is the third dataset used in our experiments, which has 279 attributes. The dataset contains a total of 13 non-empty classes. As suggested in [4],

class labels that occurred less than 5% of the dataset are considered as rare classes. The corresponding class distribution is illustrated in Table 5.

Since most attributes in this dataset are continuous, hence, we first perform a grid discretization of the data. Each attribute is divided into 2 equal-width bins.

We let each algorithm report top 85 outliers, as done in [4]. Among these reported data objects, we examine how many of them belong to rare classes. Table 6 shows the results produced by the different algorithms.

From Table 6, we can see that the algorithm in [4] produced the best result with the cost of much more running time. In the remaining algorithms, most SOE1 variations are slight better, at least achieved the same level performance. Although the performance of SOE1 algorithm on this dataset is not so good as that of the algorithm in [4], it is at least acceptable.

Table 5. Class Distribution of Arrhythmia Dataset

Case	Class codes	Percentage of instances
Commonly Occurring Classes ($\geq 5\%$)	01,02,06,10,16	92%
Rare Classes ($< 5\%$)	03,04,05,07,08,09,14,15	8%

Table 6. Detected Rare Classes in Arrhythmia Dataset

Number of Records	Number of Rare Classes Included									
	SOE1 (\prod)	SOE1 (+)	SOE1(S_q)			SOE1 (S_∞)	<i>Find</i> <i>FPOF</i>	<i>Find</i> <i>CBLOF</i>	[4]	<i>KNN</i>
			q=2	q=5	q=7					
85	33	32	33	34	33	27	32	32	43	28

5 Conclusions

From an ensemble-learning viewpoint, a unified subspace outlier ensemble framework for outlier detection in high dimensional spaces is proposed in this paper. Empirical evidence verified the feasibility and advantage of our method.

References

1. He, Z., Xu, X., Huang, J., Deng, S.: A Frequent Pattern Discovery Based Method for Outlier Detection. In: Proc. of WAIM'04, pp. 726-732, 2004
2. He, Z., Xu, X., Deng, S.: Discovering Cluster Based Local Outliers. Pattern Recognition Letters, 2003, 24(9-10): 1641-1650
3. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient Algorithms for Mining Outliers from Large Data Sets. In: Proc. of SIGMOD'00, pp. 93-104, 2000
4. Aggarwal, C., Yu, P.: Outlier Detection for High Dimensional Data. In: Proc. of SIGMOD'01, pp. 37-46, 2001
5. Merz, G., Murphy, P.: Uci Repository of Machine Learning Databases. <http://www.ics.uci.edu/mllearn/MLRepository.html>, 1996
6. Harkins, S., et al.: Outlier Detection Using Replicator Neural Networks. In: Proc. of DaWaK'02, pp. 170-180, 2002
7. He, Z., Xu, X., Huang, J., Deng, S.: Mining Class Outliers: Concepts, Algorithms and Applications in CRM. Expert System With Applications, 2004, 27(4): 681-697

A Convertible Limited Verifier Signature Scheme

Jianhong Zhang¹, Hu'an Li¹, and Jilin Wang²

¹ College of science, North China University of Technology, China
jhzhang@ncut.edu.cn

² Zhejiang University of Finance & Economics, Hangzhou, China

Abstract. In some cases, it is not necessary for anyone to be convinced a justification of signer's signature. Motivated by the above case, in this work we propose a provably secure convertible limited verifier signature scheme which based on the CDH assumption and the difficulty of solving inverse of hash function, and a prominent property of the scheme is that only the limited verifier can independently convert the limited verifier signature into the ordinary signature in the conversion phase, the original signer cannot convert signature instead of the limited verifier. The limited verifier signature schemes available have not this property. To the best of my knowledge, it seems to be first provably secure convertible limited verifier signature scheme without using random oracle model.

1 Introduction

In the ordinary digital signature schemes, anyone can verify signature with signer's public key. However, in some case, it is not necessary for anyone to be convinced a justification of signer's dishonorable message such as a bill. It is enough for a receiver only to prove a justification of the signature if the signer does not execute a contract. Undeniable signature scheme[2], the designated verifier signature[6] and the limited verifier signature[1] can solve this problem above. However, the undeniable signatures are only verified with the cooperation of the signer. It is very inconvenient and impractical in real life, for example, the signer should be unavailable, or refuse to verify. The designated verifier signature solves the above problem, but the designated verifier can not transfer the signature to convince any third party in the designated verifier signature scheme. To overcome the above two problems, we propose a novel scheme: the limited verifier signature scheme .

As for the limited verifier signature, a typical application of the scheme is the case in which a receiver and a signer are a public office and a public officer, respectively, and a signer must sign an official document that will be published after the legal time. In such a situation, our proposed limited verifier signature scheme will be shown to be more efficient than undeniable signature schemes with respect to computation and application cost. In 1999, S.Araki *et al* first gave convertible limited verifier signature scheme. Unfortunately, F.G. Zhang [4] showed that the scheme was universally forgeable, and everyone can obtain the original signature from the converted signature in S.Araki *et al* scheme. It might result in information leak.

Our Contribution: In this work, we propose a novel convertible limited verifier signature scheme based on the CDH assumption and the difficulty of solving inverse of hash function, and give proof security. Moreover, the conversion of the proposed scheme doesn't need the cooperation of the original signer. The scheme has a standout property that only the limited verifier can convert the limited verifier signature into an ordinary signature, even if the original signer cannot also convert the original signature instead of the limited verifier. Due to the property, our converted signature has intrinsically the confirmation function, namely, when the signature need to be confirmed, if the limited verifier can correctly convert the original signature, then it means that the signature was produced by the original signer. When the original signer violated the rules, the limited verifier can provide a converted signature to convince a Judge that the original signature was indeed produced by the original signer.

2 Preliminaries

Definition 1. (Computational Diffie-Hellman (CDH) assumption)

\forall PPT (Probability Polynomial Time) algorithm A

$$Pr[A(g^a, g^b) = g^{ab} | p = 2q+1, p, q \text{ prime}, a, b \leftarrow_R Z_q, g \text{ generator of } G] = \text{neg}(k)$$

(where A also knows the public modulo p and the generator $g, |p| = k$)

In other words, the CDH assumption states the fact: "Given g^a, g^b , it is hard to completely compute g^{ab} ".

Definition 2. The Difficulty of Inverting Hash Function: Given a hash function H , it don't exist a probabilistic polynomial-time algorithm A , on input y to find an inverse x' of y to satisfy $H(x') = y$ in non-negligible probability. In other words, given a hash value y , there don't exist an algorithm A that can output a number x' satisfying $H(x') = y$ in polynomial time.

Definition 3. Signature of Knowledge: Let $y, g \in G$. A pair $(c, s) \in \{0, 1\}^k \times \pm\{0, 1\}^{\epsilon(l_G+k)+1}$ verifying $c = H(y, g, g^s y^c, m)$ is a signature of knowledge of the discrete logarithm of $y = g^x$ w.r.t. base g , on a message $m \in \{0, 1\}^*$. We denote the signature by

$$SPK[\alpha : y = g^\alpha](m)$$

Definition 4. The convertible limited verifier signature scheme

- **System Parameters Generation:** Input a security parameter l , outputs the system parameters SP .
- **Key Generation:** Input the system parameter SP , outputs public/private key pair (pk_s, sk_s) and (pk_v, sk_v) of the signer and the limited verifier.
- **Signing:** Input the key pair (pk_s, sk_s) of the signer, message M and the public key pk_v of the limited verifier, outputs a limited verifier signature s .
- **The First Signature Verifying:** A determined algorithm, on input the key pair (pk_v, sk_v) of the limited verifier, the public key pk_s of the original signer, and the signature s , outputs True or False.

- **Converting phase:** on input the private key sk_v of the limited verifier , and the signature s , output the converted signature s' . Note that the signature conversion only is only executed by the limited verifier, the original signer cannot execute it, or else, if the original signer is dishonest, he can convert the signature s ahead of legal time.
- **Public Verification:** on input the public key pk_s, pk_v of the original signer and the limited verifier and the ordinary signature s' , outputs True or False.

if the proposed signature scheme satisfies the requirements except for Unforgeability and Verifiability , it is called *secure*.

- **Converted unforgeability:** after the limited signature was converted into the ordinary signature, no body can forge a valid converted signature.
- **Non-replaceability** only the limited verifier can independently convert the limited verifier signature into the ordinary signature. however, the original signer do not replace the limited verifier to convert the limited signature.

3 A Novel Limited Verifier Signature Scheme

Our proposed scheme mainly consists of system parameters Setup, the Signing phase, the first verification phase, the converted signature phase and the second verification phase. The detail procedure is as follows:

[System parameter setup] Let p, q be two large prime numbers, and satisfy $p = 2q + 1$, g is a generator of order q and $g \in Z_p^*$, $h()$ is collision-resistant hash function, for example (SHA-1,MD5).

The original signer U_A randomly chooses a number $x_A \in Z_q^*$ as his privacy key with $x_A \neq q - 1$, and computes the corresponding public key $y_A = g^{x_A} \pmod p$. A limited verifier U_B randomly chooses a number $x_B \in Z_q^*$ as his privacy key and computes the corresponding public key $y_B = g^{x_B} \pmod p$; m denotes the signed message.

[Signing phase]: To produce a signature on message m , the signer U_A computes as follows:

- Step 1: He randomly chooses a number $k \in Z_q^*$.
- Step 2: he computes

$$J = g^k \pmod p, r_1 = k + x_A J \pmod q, r_2 = h(m)(y_B^{r_1} + g)^{-1} \pmod p$$

$$s = (r_2 k - 1 - r_2 h(r_2, J))(1 + x_A h(J, r_2))^{-1} \pmod q$$

then the resultant signature is (r_2, s, J) . To be secure transmission, the original signer can encrypt the signature with the public key of the limited verifier.

[First verifying phase] : Given a signature (r_2, s, J) from the signer U_A , U_B need verify whether the following equation is valid.

$$h(m) = (y_B^{(1+r_2 h(r_2, J)+s)r_2^{-1}} (y_A^{sr_2^{-1} h(J, r_2)+J})^{x_B} + g)r_2 \tag{1}$$

if the signature (r_2, s, J) satisfies the equation (1), it means that the signature is valid, or else, it is invalid.

[Converting phase and the second verification]: When confirming the signature or after the legal time, the limited verifier can convert the original signature produced by the signer U_A into an ordinary digital signature, so that the signature can be publicly verified, Namely, everyone can verify the converted signature valid without a cooperation of the signer or the limited verifier. The conversion of the limited verifier is as follows:

1. randomly chooses a number $\lambda \in Z_q$, and computes $y_l = y_A^\lambda \pmod p$ and the knowledge signature $(\tilde{c}, \tilde{s}) = SPK[\alpha : y_l = y_A^\alpha]''$.
2. $\alpha = (r_2 h(r_2, J) + s)r_2^{-1} \pmod q$, $\beta = (sr_2^{-1} h(J, r_2) + J)x_B - \lambda y_l \pmod q$

and publishes the converted signature $(m, \alpha, \beta, r_2, y_l, (\tilde{c}, \tilde{s}))$; after arbitrary one obtains the convertible signature $(m, \alpha, \beta, r_2, y_l, (\tilde{c}, \tilde{s}))$, he first verifies the knowledge signature (\tilde{c}, \tilde{s}) validity, then he verifies whether it is valid by the following equation.

$$h(m) = (y_B^{\alpha+r_2^{-1}} y_A^\beta y_l^{y_l} + g)r_2 \pmod p \tag{2}$$

To briefly discuss, we don't consider the knowledge signature (\tilde{c}, \tilde{s}) in the following security analysis. Without loss of the generality, we assume that y_l is the format of the exponential value of y_A . Then the converted signature $(m, \alpha, \beta, r_2, y_l, (\tilde{c}, \tilde{s}))$ can be briefly denoted as $(m, \alpha, \beta, r_2, y_l)$.

4 Security Analysis

In this section, we give security analysis to our proposed signature scheme, and show that security of the scheme is based on the CDH assumption and the difficulty of solving the inverse of hash function.

Theorem 1. *A signature (r_2, s, J) produced on message m by the honest signer must pass the verifying equation (1).*

Proof. It is obvious.

According to Theorem 1, we know a signature produced by an honest signer must pass verification of the limited verifier. Thereby, it guarantees correctness of our scheme. We demonstrate that our proposed scheme is provably secure by the following several theorems.

Theorem 2. *if an adversary can forge a signature (r_2, s, J) on message m in polynomial time in non-negligible advantage $Adv(\epsilon)$, then the adversary can break the CDH assumption in polynomial time in non-negligible advantage $Adv(\epsilon)$.*

Proof. Suppose there exists an adversary A to forge a signature (r_2, s, J) on the message m in polynomial time in non-negligible probability $Adv(\epsilon)$, and then there is an adversary A' which can break the CDH assumption in polynomial time by performing subroutine A .

Assume that there is an algorithm A (oracle) which takes (y_A, y_B, m') as input and outputs a valid signature (r'_2, s', J') on a message m' which satisfies the verification equation of the limited verifier, and then the adversary A' can break the CDH assumption in polynomial time in non-negligible advantage $Adv(\epsilon)$.

Let (r'_2, s', J') be a forging signature on message m' , then the forging signature should satisfy

$$h(m') = (y_B^{(1+r'_2 h(r'_2, s') + s') r'_2{}^{-1}} (y_A^{s' r'_2{}^{-1} h(J', r'_2) + J'})^{x_B} + g) r'_2$$

from the above equation, we have

$$(h(m') r'_2{}^{-1} - g) / (y_B^{(1+r'_2 h(r'_2, s') + s') r'_2{}^{-1}}) = (y_A^{s' r'_2{}^{-1} h(J', r'_2) + J'})^{x_B} \tag{3}$$

it means that this adversary can compute

$$g^{x_A x_B} = y_A^{x_B} = \left(\frac{h(m') r'_2{}^{-1} - g}{y_B^{(1+r'_2 h(r'_2, s') + s') r'_2{}^{-1}}} \right)^{(s' r'_2{}^{-1} h(J', r'_2) + J')^{-1}}$$

Thus, Given $(y_A = g^{x_A}, y_B = g^{x_B})$, the adversary can compute $g^{x_A x_B}$ in non-negligible advantage $Adv(\epsilon)$. Then he can break CDH assumption. Obviously, it is in contradiction to the difficulty of the CDH assumption.

Remark 1. suppose an adversary Adv can forge a converted signature $(m', \alpha', \beta', r'_2, y'_i)$, and then the signature should satisfy

$$h(m') = (y_B^{\alpha' + r'_2{}^{-1}} y_A^{\beta'} y_i^{y'_i} + g) r'_2 \tag{4}$$

we know the adversary Adv can compute the right side of the equation (4), then it means that he can solve the inverse of hash function. Namely, find a message m' satisfying

$$h(m') = (y_B^{\alpha' + r'_2{}^{-1}} y_A^{\beta'} y_i^{y'_i} + g) r'_2$$

It is in contradiction with the difficulty of solving the inverse of hash function. if he chooses $(m', \alpha', \beta', r'_2)$ to compute y' satisfying the verification equation, it is equivalent to solving the difficulty of the discrete logarithm problem.

Theorem 3. *the original signer cannot produce a converted signature $(m, \alpha, \beta, r_2, y_l)$ instead of the limited verifier by his signature (r_2, J, s) in polynomial time in non-negligible probability ϵ .*

Proof. Though the signature (r_2, J, s) is produced by the signer, and he can also compute $\alpha = (r_2 h(r_2, s) + s) r_2^{-1}$, he cannot produce a converted signature instead of the limited verifier.

Suppose the original signer can produce a converted signature $(m, \alpha', \beta', r'_2, y'_i)$ on message m . Because the signature (r_2, J, s) is produced by him, and he can also compute $\alpha = (r_2 h(r_2, s) + s) r_2^{-1}$ by the signature (r_2, J, s) , obviously, $\alpha' =$

$\alpha, r'_2 = r_2$, where α, r_2 are components of the signature $(m, \alpha, \beta, r_2, y_l)$ converted by the limited verifier; then the signature $(m, \alpha', \beta', r'_2, y'_l)$ should satisfy

$$h(m) = (y_B^{\alpha' + r_2'^{-1}} y_A^{\beta'} y_l^{y'_l} + g)r_2'$$

Hence, we have

$$((h(m)r_2'^{-1} - g)/(y_B^{\alpha' + r_2'^{-1}})) = y_A^{\beta'} y_l^{y'_l} = y_A^{\beta' + \lambda' y'_l} \quad (5)$$

Whereas, we know that the signature $(m, \alpha, \beta, r_2, y_l)$ converted by the limited verifier satisfies

$$(h(m)r_2^{-1} - g)/(y_B^{\alpha + r_2^{-1}}) = y_A^{\beta} y_l^{y_l} = y_A^{(sr_2^{-1}h(J, r_2) + J)x_B} \quad (6)$$

Because of $\alpha' = \alpha, r'_2 = r_2$, it means that $(sr_2^{-1}h(J, r_2) + J)x_B = \beta' + \lambda' y'_l$ from the above equation (5) and (6), thus the original signer can compute the privacy key of the limited verifier:

$$x_B \equiv (\beta' + \lambda' y'_l)/(sr_2^{-1}h(J, r_2) + J) \text{ mod } q$$

Obviously, he can solve the discrete logarithm x_B of y_B with respect to g in polynomial time in non-negligible probability. It is in contradiction with the difficulty of solving the discrete logarithm problem in finite field. Hence, it satisfies the non-replaceability.

5 Conclusion

In some cases, we needn't be any one to verify signature valid. In this paper, we proposed a provably secure convertible limited verifier signature scheme without the random oracle model. Security of the scheme is based on the CDH assumption and the difficulty of solving inversion of hash function. The limited verifier can independently convert the original signature into the ordinary without the help of the original signer. It is open problem to design a designated oriented-group limited verifier signature scheme.

References

1. S. Araki, S. Uehara and K. Imamura, The limited verifier signature and its application, IEICE Transactions on Fundamentals E82-A (1), pp. 63-68, 1999.
2. J. Boyar mgard and T. Pedersen "Convertible undeniable signatures" Advances in Cryptology -CRYPTO'90, Springer-Verlag, 1990, pp.189-205.
3. D. Chaum and H. van Antwerpen "Undeniable signatures," Advances in Cryptology-CRYPTO'89 Springer-Verlag, 1990, pp.212-216.
4. Fangguo Zhang and Kwangjo Kim, "A Universal Forgery of Araki et al.'s Convertible limited Verifier Signature Scheme", IEICE Trans. Fundamentals, Vol.E86-A, No.2, pp. 515-516, 2003.

5. A.Araki, S.Uehara, and K.Imamura, "Convertible limited verifier signature based on Horster's authenticated encryption," 1998 Symposium on Cryptography and Information Security, SCIS'98, 1998
6. M. Jakobsson, K. Sako, and R. Impagliazzo, " Designated verifier proofs and their applications," Advances in Cryptology-Eurocrypt 1996, LNCS 1070, pp.143-154, Springer- Verlag, 1996.

Short Signature Scheme Based on Discrete Logarithms

Zuhua Shao

Zhejiang University of Science and Technology,
No. 85, XueYuan Road, Hangzhou,
Zhejiang, 310012, P.R. of China
zhshao_98@yahoo.com

Abstract. This paper proposes a short signature scheme, the security of which is based on the hardness of discrete logarithms. The main advantage of this signature scheme over DSA signature schemes is a one fourth reduction in the signature length, as well as in the verification computation. Moreover, we provide a close reductionist security proof for existential unforgeability under adaptive chosen-message attacks in random oracle model, offering better security guarantees than existing discrete-log based signatures. The new scheme is needed in low-bandwidth communication, low-storage and less computation environments, and is particularly suited for smart card, PDA and mobile phone.

Keywords: Short signature, discrete logarithm, random oracle model, reductionist security proof.

1 Introduction

Short digital signatures are needed in low-bandwidth communication and low-storage and less computation environments. For example, short signatures are needed when printing a signature on a postage stamp, a commerce involve or a bank bill. Short digital signatures are also needed where a human is asked to manually key in signatures. For instance, product registration systems often ask users to key in a signature provided on a CD label.

With the popularizing of Internet and E-commerce, it has been attractive to construct a digital signature scheme with both shorter signature length and less computation.

A number of short signature schemes have been proposed so far. Some proposals tried to shorten RSA based signatures. Other proposals showed how to shorten the discrete-log based signature schemes while preserving the same level of security. Naccache and stern [1] proposed a variant of DSA to sign on postcard, where the signature length is approximately 240 bits. The technique proposed for reducing the DSA signature length is signatures with message recovery. In such

systems, one encodes a part of the message into signatures, thus shortening the total length of the message-signature pair. For long messages, one can then achieve a DSA signature overhead of length 160 bits. However, for very short message (e. g., 64 bits) the total length is still 320 bits. Moreover, when messages are not transmitted, DSA signatures with message recovery are not any shorter than standard DSA signatures.

The shortest signature known in the classical cryptography is based on Weil pairing and achieves 160 bits with the security level of 2^{80} . Boneh et al. [2] introduced a short signature scheme (BLS) from the Weil pairing based the computational Diffie-Hellman assumption on certain elliptic curves and hyper-elliptic curves. Later, Boneh et al. [3] also proposed a new short signature scheme where signatures are almost as short as BLS signature scheme without random oracle model under a strong Diffie-Hellman problem assumption. Zhang et al. [4] improved BLS scheme by replacing special hash functions by general cryptography hash functions such as SHA-1 [5] or MD5. However, the evaluation of bilinear pairings used by pairing based short signature schemes is more time-consuming. They are not as efficient as DSA type signature schemes in terms of computation. Hence the storage efficiency of pairing based signatures is at cost of sacrificing their computation efficiency.

Besides the signature length of discrete-log signature schemes, other problem is the loose security related to the underlying hard computational problem. Some discrete-log based signature schemes, such as ElGamal and DSA, require non-standard security assumptions. Other schemes, such as Schnorr signature, the “modified ElGamal” signature of Pointcheval and Stern [6], and some DSA variants, have security proofs that are only loosely related to discrete-log problems. The only known reduction converting a forging algorithm for a discrete-log signature scheme into an algorithm that breaks the underlying discrete-log problem is the “forking lemma”. However, this reduction is inefficient to solve the discrete-log problem with a probability comparable to the success probability of the signature forger. Goh and Jarecki [7] proposed a signature scheme whose security is tightly related to the Computational Diffie-Hellman (CDH) assumption in random oracle model. Though the hardness of the CDH is widely believed to be closely related to the hardness of the DL problem [8], the latter is certainly not easy than the former.

In this paper, we would like to propose a new short signature scheme SDL based on discrete-log. The main advantage of this signature scheme over DSA signature schemes is a one fourth reduction in the signature length, as well as in the verification computation. Moreover, we will show that the security of the proposed signature scheme is close, if not tightly, related to the hardness of discrete logarithms, by providing an efficient reductionist security proof for existential unforgeability under adaptive chosen-message attacks in random oracle model, offering better security guarantees than existing discrete-log based signatures. The new scheme is needed in low-bandwidth communication, low-storage and less computation environments.

2 The Proposed Short Signature Scheme

Let p and q be large primes with $q|(p-1)$. Also let $G_{p,q} = \{g^0, g^1, \dots, g^{q-1}\}$ be a prime order q subgroup of the multiplicative group Z_p^* , where g is a generator with prime order q . Let H and F be (ideal) hash functions where

$$H: \{0,1\}^* \times Z_p^* \rightarrow \{0,1\}^{n_q/2} \quad \text{and} \quad F: \{0,1\}^* \rightarrow Z_p^*$$

For the notational convenience, we will omit the “(mod p)” and “(mod q)” markers. We denote the bit length of q by $|q| = n_q$ and that of p by $|p| = n_p$. The notation $a \xleftarrow{R} S$ means that a is picked uniformly at random from a set S .

We now describe the SDL signature scheme in full detail.

- The key generation algorithm **Gen**: Pick a random $x \xleftarrow{R} Z_q^*$ as the private key. The corresponding public key is $y \leftarrow g^x$.
- The signing algorithm **Sign**: The inputs are the private key x , the public key y and a message $m \in \{0,1\}^*$. First pick a random $k \xleftarrow{R} Z_q^*$. Compute $f \leftarrow F(m)$, $r \leftarrow g^{kf}$, $h \leftarrow H(m, r)$ and $s \leftarrow k - hx$. The signature of the message m is $\sigma \leftarrow (h, s)$.
- The verification algorithm **Ver**: The inputs are the public key y and the message m and the signature $\sigma = (h, s)$. First compute $f \leftarrow F(m)$, $r' \leftarrow fy^h g^s$ and $h' \leftarrow H(m, r')$. If $h = h'$ output *valid*, otherwise output *invalid*.

Completeness: Because $s = k - hx$ and $r = g^{kf}$, $k = s + hx$ implies $y^h g^s = g^k$ and $r' = fy^h g^s = g^{kf} = r$. So $h' = H(m, r') = H(m, r) = h$. Hence, the signature $\sigma = (h, s)$ produced by the signing algorithm **Sign** is always *valid*.

Note that our results can also be carried over to other groups, such as those built on elliptic curves.

3 Security Proof of the Proposed Signature Scheme

In fact, the proposed short signature scheme SDL is a variant of Schnorr signature scheme. This is a generic digital signature schemes considered by Pointcheval and Stern [9] which, given the input message m , produce triples (σ_1, h, σ_2) where σ_1 randomly takes its value in a large set, h is the hash value $(H(m, \sigma_1), F(m))$ and σ_2 only depends on σ_1 , the message m , and h .

By directly following the technique of Pointcheval and Stern, we can obtain the generic result:

Theorem 1 (The Forking Lemma): Let A be a probabilistic polynomial time Turing machine whose input only consists of public data. We denote respectively by Q and R

the number of queries that A can ask to the random oracle and the number of queries that A can ask to the signer. Assume that, within a time bound T , A produces, with probability $\varepsilon \geq 10(R+1)(R+Q)2^k$, a valid signature $(m, \sigma_1, h, \sigma_2)$. If the triple (σ_1, h, σ_2) can be simulated without knowing the secret key, with an indistinguishable distribution probability, then there is another machine which has control over the machine obtained from A replacing interaction with the signer by simulation and produces two valid signatures $(m, \sigma_1, h, \sigma_2)$ and $(m, \sigma_1, h', \sigma_2')$ such that $(H(m, \sigma_1), F(m)) \neq (H'(m, \sigma_1), F(m))$ in expected time $T \leq 120686T/\varepsilon$.

In the notation of our scheme, we obtain two equations:

$$F(m)g^s y^{H(m,r)} = r \text{ and } F(m)g^{s'} y^{H'(m,r)} = r$$

From them, we can compute the secret value $x = (s - s')/(H'(m, r) - H(m, r))$.

However, the reductionist algorithm by the Forking Lemma is not efficient, so that the relation between the security of the short signature scheme SDL and the hardness of the DL problem is loose.

Intuitively, if the simulator answers the query $F(m)$ by rg^t , instead of a random number t in Z_p^* (Note that the simulator answers the F -query $F(m)$ with random number t in Z_p^* in the proof of Theorem 1), we would obtain the secret value $x = (-s - t)/(H(m, r))$. Thus, the oracle replay attack is no longer required.

Hence, we would like to present a more efficient reductionist algorithm in detail. The close relation between the security of the short signature scheme SDL and the hardness of the DL problem is provided by the following theorem:

Theorem 2: Let $G_{g,p}$ be a (t', ε') -DL group, then the short signature scheme is $(t, q_H, q_F, q_{sig}, \varepsilon)$ -secure against existential forgery on adaptive chosen message attack in the random oracle model, where

$$t' \approx (3/2)(t + (2q_s + q_F)C_{exp}(G_{g,p}))$$

$$\varepsilon' \approx \frac{\varepsilon}{2} + \frac{\varepsilon^3}{8} \left(\frac{1}{2} + \frac{1}{q_H} \right) - 2^{-n_q/2} - q_S(q_S + q_H)2^{-n_p}$$

Here $C_{exp}(G_{g,p})$ denotes the computation of a long exponentiation in the group $G_{g,p}$. For the reason of page limits, the proof is written in the full paper version.

The Security of the Hash Functions: To obtain a short signature, we let q be 160 bits. It is easy to find messages m and m' such that $H(m, r) = H(m', r)$ by birthday attacks since the hash function value of H is 80 bits. If the attacker asks for a signature on message m , the signature the signer returns is based on an arbitrary number r' rather than r . Though, we are not sure that it is infeasible to find other message m' with $H(m', r') = H(m, r')$, we are sure that it is infeasible to find m' with $F(m') = F(m)$ since the hash function value of F is at least 1024 bits.

Meanwhile no method has been found to find s, h from the multivariant congruence $H(m, F(m)g^s y^h) = h$ or find r, s from $F(m)g^s y^{H(m, r)} = r$. Because random oracle models assume that the hash functions are ideal, the probability $\Pr_{r \in Z_p} \{H(m, r) = h \mid \forall m \in \{0,1\}^*, \forall h \in 2^{n_q/2}\} = 2^{-n_q/2}$.

4 Conclusions

We have proposed a short signature scheme SDL, the security of which is based on discrete logarithms. We have provide a close reductionist security proof for existential unforgeability under adaptive chosen-message attacks in random oracle model, offering better security guarantees than the existing discrete-log based signatures. In practice, the hash function H and F are not really random. Thus the security proof in the random oracle model is no longer valid. Nevertheless, this security proof does guarantee security against an adversary who does not exploit any property whatsoever of the hash functions H and F .

The contribution of the short signature scheme over DSS is a one fourth reduction in the bit size of the signature, as well as in the verification computation, since the computation time of the exponentiation is proportional to the bit size of the exponent. The new scheme is more suitable for low-bandwidth communication, low-storage and less computation environments, and is particularly suited for smart card, PDA and mobile phone.

Recently, Biham and Chen [10] found two near-collisions of the full compression function of SHA-0 and millions of collision of reduced variant of SHA-1. Wang⁶⁹ and her colleagues further broke full SHA-1 by using 2^{80} steps, much faster than 2^{80} steps of birthday attack. These breaking progressions compel us to reconsider the security of hash functions which are indispensable in signature schemes. To guarantee security, we should use longer hash functions, such as SHA-256, SHA-512, which would increase the order of the subgroup of the multiplicative group Z_p^* , thereby lengthen the signatures of DSA, Schnorr and BSL et al. schemes and increase the computation load of these signature schemes. However, the proposed short signature scheme SDL maintains the *status quo* since it does suffer from these problems. Therefore our SDL signature scheme is advantageous over the extended DSA, Schnorr and BSL et al. schemes in future.

Acknowledgements

This material is based upon work funded by Zhejiang Provincial Natural Science Foundation of China under Grant No.Y104201.

References

1. D. Naccache, J. Stern, Signing on a Postcard, In *Proceedings of financial Cryptography'00*, LNCS 1962, pp. 121-135, Springer-Verlag, (2001).
2. D. Boneh, B. Lynn, H. Shacham, Short signature from the Weil pairing, In *Proceeding of Asiacrypt'01*, LNCS 2248, pp.514-532, Springer-Verlag, (2001).

3. D. Boneh, X. Boyen, Short signature without random oracles, In *Proceedings of EUROCRYPT'04*, LNCS 3027, pp. 56-73, Springer-Verlag, (2004).
4. F. Zhang, R. Safavi-Naini, W. Susilo, An efficient signature scheme from Bilinear pairings and Its application, In *Proceedings of PKC 2004*, LNCS 2947, pp. 277-290, Springer-Verlag, (2004).
5. NIST. Secure Hash Function (SHS). Publication 180-1, *Federal Information Processing Standards*, April (1995).
6. D. Pointcheval, J. Stern, Security proofs for signature schemes, In *Proceedings of Eurocrypt'96*, LNCS 1070, pp.387-398, Springer-Verlag, (1996).
7. E. -J. Goh, S. Jarecki, A signature scheme as secure as the Diffie-Hellman problem, In *Proceedings of Eurocrypt'03*, LNCS 2656, pp. 401 – 415, Springer-Verlag, (2003).
8. U. Maurer, S. Wolf, The relationship between breacking the Diffie-Hellman protocol and computing discrete logarithms, *SIAM Journal on Computing*, 28(5), pp.1689-1721, (1999).
9. D. Pointcheval, J. Stern, Security arguments for digital signatures and blind signatures, *Journal of Cryptology*, vol. 13, No. 3, pp. 361-396, Mar, (2000).
10. E. Biham, R. Chen, Near-Collisions of SHA-0, In *Proceedings of Crypto'03*, LNCS 3152, pp. 290 – 315, Springer-Verlag, (2004).

Simulating a Finite State Mobile Agent System

Liu Yong, Xu Congfu, Chen Yanyu, and Pan Yunhe

College of Computer Science, Zhejiang University,
Hangzhou 310027, P.R. China

Abstract. This paper analyzes the simulation of Finite State mobile agent, which regards the migration, execution, and searching of the mobile agents as finite states. As the finite state mobile agents are deployed under the virtual organization platform, the corresponding search algorithm in virtual organization is also introduced. The experiments results based on the finite mobile agent's evaluation model show the finite mobile agents can achieve satisfied performance.

1 Introduction

A mobile agent is a program that can move through a network under its own control, capable of navigating through the underlying network and performing various tasks at each node independently [10]. Mobile agents are an effective paradigm for distributed applications, and are particularly attractive in a dynamic network environment involving partially connected computing elements [11]. They locate for the appropriate computation resources, information resources and network resources, combining these resources in a certain host, to achieve the computing tasks. So the mobile agent is the desirable technology for the next generation network applications. While mobile agents have generated considerable excitement in the research community, and also have many prototype systems [8], [9], they have not translated into a significant number of real-world applications. One of the main reasons for this is the lack of work that quantitatively evaluates. The works on the mobile agents performance evaluation have attracted many researchers [2], [3], [4], [1]. Strasser and Schwehm [2] develop a general model for comparing the performance of Remote Procedure Calls (RPC) with the performance of migrating agents. Johansen [4] shows that MAs perform better as the size of the data increase, and as the number of clients increase. More recently, Puliafito et al. [3] used Petri nets to compare the mobile-agent, remote-evaluation and client-server paradigms. David Kotz etc [1] develop an analytical model in wireless network. These papers propose a number of performance benefits that can be gained through the use of MAs.

The Grid technology [5] and the VO (virtual organization) technology [6] provide a new approach for the self-organization and self-management among the network nodes, and also provide a powerful platform for the mobile agent. The virtual organization can help the mobile agent to ignore the different OS problem and communication problem, decrease the complexity of the mobile agent greatly. Although the performance analysis on mobile agents has been studied for several

years, to our best knowledge, not much work has yet been done to the mobile agents' performance evaluation on VO platforms.

In this paper, we present a Finite State Mobile Agent (FS-MA) [7] performance evaluation experiment in the VO platforms, in our work, we focus on quantitative performance evaluation of FS-MA on VO platforms, and propose a framework for investigating the performance characteristics of FS-MA platforms and applications. Experimental results provide us with initial conclusions that lead to further refinement and extension of FS-MA and benchmarks.

2 VO Based FS-MA and MACM

2.1 VO Based Finite State Mobile Agents

In our Finite state mobile Agent model, there is fabric architecture, named virtual organization (VO or group), to support the computation. The basic elements of virtual organization are nodes, which can be PC, PDA, laptop, and other devices that connect via network. The nodes group in virtual, and they can join and leave the group dynamically. The groups are virtually hierarchical, which means the groups can be classified into root-layer, middle-layers and leaf virtual layers. The virtual group based fabric architecture is the platform of the mobile agent migration. By this way, the mobile agent can discovery and move more effective and it also can greatly decrease the mobile agent size when migration.

Finite-state mobile agent is a resource driven mobile agent system. In fact, the mobile agent can be seen as a finite-state machine auto motioning and driven by the resource and data. The FS-MA is a finite state machine driven by the resource and service time. And the migration state between the block state and the serve state will insure that the agent can find and move to another node that takes on enough resource when there is not enough executing resource in the local node.

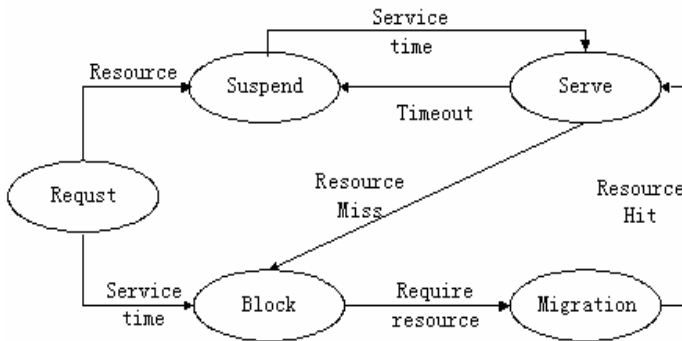


Fig. 1. Transition Relationship of the Finite State Agent

2.2 Mobile Agent Computation Model (MACM)

In practice, the topology of network is unstable, and the resources in those nodes of network are consumed and regenerated continually. So the executing routes of the mobile agents are different. We use service computing finite state machine to describe the real executing process of the service under the VO [6]. Because the service computing is the executing process of the mobile agent, the service computing finite state machine is also called MACM, mobile agent computing model finite state.

The definition of the MACM-FS establishes upon the virtual origination [6] and the definition is presented as follows:

Mobile agent computation model MACM is an eighttuple

$MACM = (R, S, M, \Phi, v, E, \Lambda, \mathcal{E})$, where R is the node set, S is the finite state set of the mobile agent, it refers the service process of the agents. S does not include the state of the agent migration Λ and the null state \mathcal{E} . Here, migration state Λ means that the mobile agent starts moving to another node to execute new state; null state \mathcal{E} means the mobile agent does not perform any action (executing and migration), M is the set of all the message operation states for mobile agent. $M = \{M_s, M_a\}$, M_s is the state of sending message, M_a is the state of receive message, $v \in R$ is the initial node that the mobile agent has been produced, a mobile agent's service firstly comes from the node v , and then cycles driven by the finite states, $E \subset R$ is the set of final node for the mobile agent, only in the final node the mobile agent can be destroyed and the service ends, Φ , The transition relation, is a finite subset of $(R \times (S \cup \{\Lambda, \mathcal{E}\})) \rightarrow R$:

$\Phi : R \times (S \cup \{\Lambda, \mathcal{E}\}) \rightarrow R$, where

- (1) To all the $R_i, R_j \in R$, if $\Phi(R_i, \mathcal{E}) = R_j$, then $R_i = R_j$,
- (2) To all the $R_i, R_j \in R$, if $\Phi(R_i, \Lambda) = R_j$, then $R_i \neq R_j$,
- (3) To all the $R_i, R_j \in R, S_k \in S$, if $\Phi(R_i, S_k) = R_j$, then $R_i = R_j$,
- (4) To all the $R_i \in R$, if $\Phi(R_i, M_a) = R_i$, then the next transition state relation is $\Phi(R_i, M_s) = R_i$.

In this computation model the migration state Λ is established by the communication of the nodes in VO. Using the communication algorithm, mobile agent can move from the original node to the destination node efficiently. The state transition and message communication are both implemented by this algorithm.

3 Evaluation of the Finite State Mobile Agent Computation Model

To study MA system performance, one should take into account issues such as: control and state information; the complicate architecture of MA platforms; the variety of distributed computing (software) models applicable to mobile-agent applications; the continuously changing resource configuration of Internet-based systems.

VO [6] is a virtual and dynamic hierarchical architecture in which Grid nodes are grouped virtually. Nodes can join the group and leave the group dynamically. The groups are virtually hierarchical, with one root-layer, several middle-layers, and many of leaf virtual groups. Among each leaf group, one (just one) node (called as manger or gateway node) is chosen to form upper-layer groups in the same way, and this way is repeated until to form one root layer group. Gateway nodes will forward the low-layer group's status information to all the nodes in the up-layer group, and distribute the upper-layer group's status information to all the nodes in the lower-layer group.

We simulate the VO architecture on PC systems and make it self-grouping dynamically according to Grid size, the number of nodes in Grid and the resources on nodes. Every node can search resources in Grid, deploy FS-MA, and communicate with other node through VO. In fact, the communications are implemented by the group's gateway node, which is similar to the gateway in TCP/IP protocol.

There are two important factors must be considered carefully when grouping the Grid nodes. One is the distance between the nodes and group manager; the other is the resource distribution among the nodes. In our evaluation model, we use a new concept named Resource Homologies to describe the resource distribution among nodes. Normally, the resource can be viewed as a multiple-dimension vector \vec{R} , each dimension in the vector represents one kind of resource. So the Resource Homologies of the nodes A and B can be defined as $|\vec{R}_A - \vec{R}_B|$.

4 Results

In this section, we present and obtain the simulation results of the previous evaluation models for the FS-MA.

The single agent experiments focus on the agent performance on the measurement of service availability. The agent in the experiments (except Figure 5) is a five-state mobile agent, that means the agent includes five service-executing states and will request resource in the VO platforms at most five times. In the figure 2, the diagram presents the results for 1000 times of the same agent executing in different resources distributions conditions. Here, although the resources are re-distributed and the topology of the VO platform is re-generated each time, the total VO grid size and the nodes number on the VO platform are the same. The results in figure 2 show that the service availability of the agent will grow up with the average state executing time rising, and the resources and nodes distribution cannot make visible influence on the service availability. Figure 3 presents the relation between the average service availability and the average state executing time with different agent size. The results show the agent size has obvious affection on the service availability. The service availability will decrease with the agent size increasing.

As we can see from Figure 4 and Figure 5 the agent total executing time mounts up with the increasing of the average state executing time and the migration times, and also with the augment of agent size.

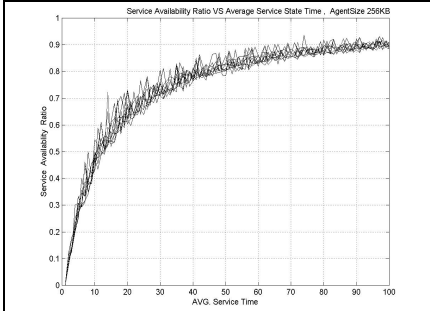


Fig. 2. service available ratio

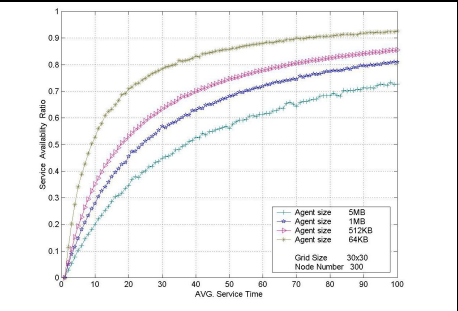


Fig. 3. average service available ratio

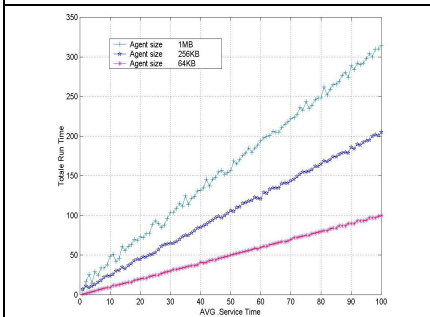


Fig. 4. average life time

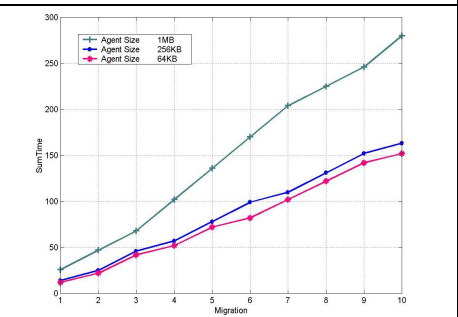


Fig. 5. life time VS migration times

5 Conclusions

In this paper, we introduce an evaluation model of the finite state mobile agent, which the executing, blocking and migration of the agents are treated as a finite state. The model is constructed on the virtual organization based platform and the corresponding search algorithm on that platform is also presented. The evaluable experiments for the single agent and multiple agents are also introduced. The experiments results indicate the finite state mobile agent can achieve stable performance in varied environments.

Future works will consider the agent system performance with more influence parameters, such the group rate of the virtual organization, size of the topology etc. The system workloads, such as the CPU workload, memory workload, and I/O workload, are also included in our future works.

Acknowledgement

This paper is sponsored by National Science Foundation of China (No.60402010) and Zhejiang Province Science Foundation (No.M603169), Advanced Research Project of China Defense Ministry (No.413150804), and partially supported by the Aerospace Research Foundation (No. 2003-HT-ZJDX-13).

References

- [1] Kotz, D., Cybenko, G., Gray, R. S., Jiang, G., Peterson, R. A. et al.. Performance analysis of mobile agents for filtering data streams on wireless networks. *Mobile Networks and Applications*, 2002, 7, 163-174
- [2] Strasser, M., Schwehm, M.. A performance model for mobile agent systems. In: *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'97)*, June 1997, Vol.2, 1132-1140
- [3] A. Puliafito, S. Riccobene and M. Scarpa, An analytical comparison of the client-server, remote evaluation and mobile agents paradigms, in: *Proceedings of the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents (ASA/MA99)* (IEEE Computer Society Press, October 1999).
- [4] Johansen, D.: Mobile Agent Applicability. In: Rothermel, K., Hohl, F. (Eds.): *Proceedings of the 2nd Int. Workshop on Mobile Agents (MA'98)*. Lecture Notes in Computer Science, Vol. 1477. Springer-Verlag, Berlin Heidelberg New York (1998) 80-98. 442, 443, 443, 444
- [5] Foster, I., Kesselman, C., Tuecke, S.. The anatomy of the grid: enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 2001, 15(3).
- [6] Huang Li-Can, Wu Zhao-Hui, Pan Yun-He. Virtual and dynamic hierarchical architecture for E-science grid. *International Journal of High Performance Computing Applications*, Volume 17 Issue 3- August 2003
- [7] Liu Yong, Xu Congfu, Wu Zhaohui et al.. A finite state mobile agent computation model. In: *Proceedings of Advanced Web Technologies and Applications: the 6th Asia-Pacific Web Conference, APWeb 2004*, Hangzhou, China, April 14-17, 2004, 152-157
- [8] Object Space. Voyager core package technical overview. <http://www.recursionsw.com/products/voyager/voyager.asp>
- [9] Lange, D., Oshima, M.. Programming mobile agents in Java - with the Java Aglet API. <http://www.cis.upenn.edu/~bcpierce/courses/629/papers/AgletsBook-index.html>
- [10] Vu Anh Pham and Ahmed Karmouch, *Mobile Software Agents: An Overview*, IEEE Communication Magazine, July, 1998.
- [11] R.Gray, D.Kotz, S.Nog and G.Cybenko, *Mobile agents for mobile computing*, Technical Report PCS-TR96-2X5, Department of Computer Science, Dartmouth College, Hanover, NH 03755, May, 1996.

Algorithm for Analyzing N-Dimensional Hilbert Curve

Chenyang Li and Yucai Feng

College of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, 430074, China
tolcy@hotmail.com

Abstract. The Hilbert curve is a way of mapping the multidimensional space into the one-dimensional space. Such mappings are of interest in a number of application domains including image processing and, more recently, in the indexing of multidimensional data. However, little has been discussed on its high dimensional algorithms due to the complexity. In this paper, a novel algorithm is presented for analyzing an N-dimensional Hilbert curve, which discusses how to obtain the constructing information of an N-dimensional Hilbert curve.

1 Introduction

In 1890, Italian mathematician G. Peano presented a family of curves, which pass through all points in a space [1]. Curves of this type have come to be called Peano curves or space-filling curves. It is well known that space-filling curves describe a method of one-to-one mapping between N-dimensional space and one-dimensional space. Such mappings are of interest in a number of application domains including image processing [2], [3] and, more recently, in the indexing of multi-dimensional data [4], [5]. Among these space-filling curves, Z-order curve [1] and Hilbert curve [6] are the most famous, for Z-order curve is the simplest in encoding and Hilbert curve achieves the best clustering [7]. For the most part, interest in applications of space-filling curves has been confined to Z-order mapping due to its simpleness. Relatively little work has been devoted to techniques for Hilbert curve in more than 2 dimensions due to the complexity. Kamata, Richard and Yukihiro presented a representative N-dimensional Hilbert mapping algorithm [8]. Its complexity is much higher than Z-order mapping, for their method has to analyze the Hilbert curve dynamically. We propose a novel algorithm for analyzing the Hilbert curve to generate a static evolvment rule table. It is the constructing information of an N-dimensional Hilbert curve, through which we can generate N-dimensional Hilbert curves and implement N-dimensional Hilbert mappings efficiently.

2 Concepts of a Hilbert Curve

Firstly, Let R^N be an N-dimensional space. The coordinates in R^N are denoted as X_0, \dots, X_2, X_1 , and the mapped coordinate value in R^1 as *Hilbert code (H-code)*. Here, we call a Hilbert curve an N-dimensional *m*th-generation Hilbert curve, if it passes through $2^m \times 2^m \times \dots \times 2^m$ (i.e., 2^{mN}) subhypercubes of a hypercube in R^N . In general it can be expressed by using H_m^N . Fig. 1 shows H_1^2 , H_2^2 and H_3^2 .

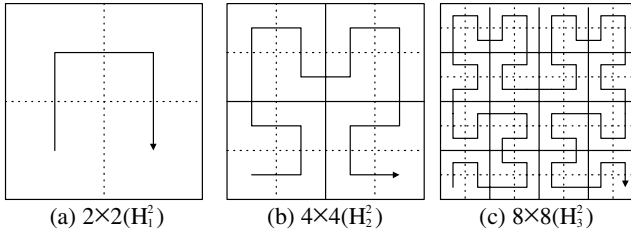


Fig. 1. Original Hilbert curves

Definition 1. (*N-dimensional Hilbert cell: C^N*)

An *N*-dimensional Hilbert cell is a *N*-dimensional *l*th-generation Hilbert curve (i.e., H_l^N).

Let C^N be an *N*-dimensional Hilbert cell. Fig. 1(a) is an example of C^2 and Fig. 2(a) is an example of C^3 .

Definition 2. (*N-dimensional Hilbert gene*)

The Hilbert gene is a list of coordinate transformation commands, which directs how to generate H_m^N from H_{m-1}^N .

Table 1 is an example of a 3-D Hilbert gene list. We describe a gene list by using $G[i]$.

Table 1. $G[i]$: 3-D Hilbert gene list ('-' denote no transformation)

Location	Exchange	Reverse
0	$X_1 \leftrightarrow X_3$	-
1	$X_2 \leftrightarrow X_3$	-
2	-	-
3	$X_1 \leftrightarrow X_3$	X_1', X_3'
4	$X_1 \leftrightarrow X_3$	-
5	-	-
6	$X_2 \leftrightarrow X_3$	X_2', X_3'
7	$X_1 \leftrightarrow X_3$	X_1', X_3'

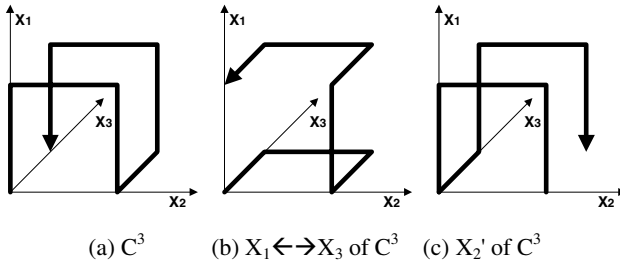


Fig. 2. Examples of coordinate transformation

The coordinate transformation includes two types: exchange ($\leftarrow\rightarrow$); reverse ($'$). Fig. 2(a) shows an initial state of C^3 , Fig. 2(b) shows the transformed result of C^3 through exchange ($X_1\leftarrow\rightarrow X_3$), and Fig. 2(c) shows the transformed result of C^3 through reverse (X_2').

3 Algorithms for Analyzing an N-Dimensional Hilbert Curve

Algorithms include generating Hilbert cell and generate the Hilbert gene.

3.1 Generating N-Dimensional Hilbert Cell

The algorithm can be described as follows:

$$\begin{cases} C^1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} & n=2, 3, \dots \\ C^n = \text{Merge} \left(\begin{matrix} 0 \oplus C^{n-1} \\ 1 \oplus \text{Reverse}(X_{n-1} \text{ of } C^{n-1}) \end{matrix} \right) \end{cases} \tag{1}$$

Here, we express a cell just using a 1-dimensional array: $C^N[\text{H-code}] = X_N \dots X_1$. H-code is the coordinates value in R^1 . $X_N \dots X_1$ is the coordinates value in R^N . For example, C^3 can be described as $C^3[0]=000$, $C^3[1]=001$, $C^3[2]=011$, $C^3[3]=010$, $C^3[4]=110$, $C^3[5]=111$, $C^3[6]=101$, $C^3[7]=100$.

3.2 Generating N-Dimensional Hilbert Gene

There are two steps to obtain the Hilbert gene: the first step is confirming the entry and exit of $H_1^N[i]$; the second step is generating $G[i]$ by analyzing the entry and exit of $H_1^N[i]$.

Confirming the Entry and Exit of $H_1^N[i]$

The algorithm for confirming the entry and exit of $H_1^N[i]$ is based on analyzing a Hilbert cell. We express the entry and exit of $H_1^N[i]$ by using two integers: $H_1^N[i][0] = X_N \dots X_1$; $H_1^N[i][1] = X_N \dots X_1$. Here, $H_1^N[i][0]$ is the entry of $H_1^N[i]$, and $H_1^N[i][1]$ is the exit of $H_1^N[i]$. And 'i' is the order in R^1 . X_N, \dots, X_1 is the coordinate values in R^N . Now, we give the algorithm for confirming the entry and exit of $H_1^N[i]$ ($i=0, \dots, 2^N-1$) as follows:

```

Confirm_Entry_Exit(input i)
{
  if (i < 2N/2) {
    if (i == 0) {
      H1N[i][0] = CN[0];
      H1N[i][1] = H1N[i][0] ^ (CN[0] ^ CN[1]);
    }
  }
  else{

```



```

H1N [i] [0]=H1N [i-1] [1] ^ (CN[i-1] ^ CN[i]);
if (H1N [i] [0] & (CN[i] ^ CN[i+1])) == CN[i] & (CN[i] ^ CN[i+1])) {
    H1N [i] [1]=H1N [i] [0] ^ (CN[i] ^ CN[i+1]);
}
else{
    H1N [i] [1] has N-1 alternative options;
}
}
else{
    if (i==2N/2) {
        H1N [i] [0]=H1N [2N-i-1] [1] ^ (0x00000001<<n-1);
        H1N [i] [1]=H1N [i] [0] ^ (H1N [2N-i-1] [0] ^ H1N [2N-i-1] [1]);
    }
    else{
        H1N [i] [0]=H1N [i-1] [1] ^ (H1N [2N-i] [0] ^ H1N [2N-i-1] [1]);
        H1N [i] [1]=H1N [i] [0] ^ (H1N [2N-i-1] [0] ^ H1N [2N-i-1] [1]);
    }
}
}
}

```

Note: The mark ‘^’ is XOR (0^1=1; 0^0=0; 1^1=0), and the mark ‘&’ is AND (0&1=0; 0&0=0; 1&1=1).

The entry and exit of H₁³[i] can be described as follows:

```

H13 [0][0]=000, H13 [0][1]=001; H13 [1][0]=000, H13 [1][1]=010;
H13 [2][0]=000, H13 [2][1]=100; H13 [3][0]=101, H13 [3][1]=100;
H13 [4][0]=000, H13 [4][1]=001; H13 [5][0]=000, H13 [5][1]=100;
H13 [6][0]=110, H13 [6][1]=100; H13 [7][0]=101, H13 [7][1]=100.

```

Generating Gene List of an N-Dimensional Hilbert Curve

Now, we give an algorithm for obtaining the gene list by analyzing the entry and exit of H₁^N [i]. Here, We express exchange transformation by using G[i][0], and express reverse transformation by using G[i][1]. The algorithm can be described as follows:

```

Confirm_GeneList(input i)
{
    G[i] [0] = ( CN[0] ^ CN[2N-1] ) ^ (H1N [i] [0] ^ H1N [i] [1]);
    G[i] [1] = CN[0] ^ H1N [i] [0];
}

```

The 3-D Hilbert gene list can be expressed as follows:

```

G[0][0]=101, G[0][1]=000; G[1][0]=110, G[1][1]=000;
G[2][0]=000, G[2][1]=000; G[3][0]=101, G[3][1]=101;
G[4][0]=101, G[4][1]=000; G[5][0]=000, G[5][1]=000;
G[6][0]=110, G[6][1]=110; G[7][0]=101, G[7][1]=101.

```

4 Growing of an N-Dimensional Hilbert Curve

It is well known that cells replicate according to the genetic information to grow into a complex organism. Generating a Hilbert curve can be described as a similar course. We use $H_m^N[i]$ to express the transformed result of H_m^N . Here ‘i’ denotes the sequence number of Hilbert order in R^1 (i.e., the location of $H_{m-1}^N[i]$ in H_m^N). Now, we take a 3-D Hilbert curve as an example to describe the course of growing of a Hilbert curve:

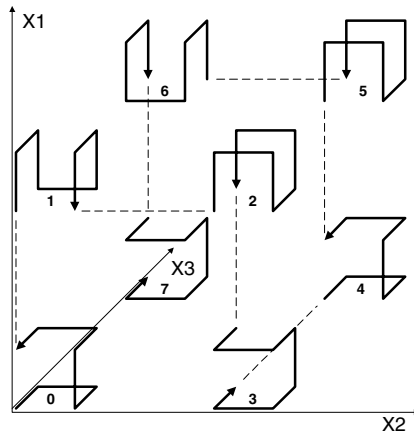


Fig. 3. H_1^3 (i.e., C^3) \rightarrow H_2^3 according to $G[i]$

Fig. 3 shows how to construct H_2^3 through assembling the eight of $H_1^3 [i]$ ($i = 0, 1, \dots, 7$) : Generating $H_1^3 [0]$ through $X_1 \leftrightarrow X_3$; Generating $H_1^3 [1]$ through $X_2 \leftrightarrow X_3$; Generating $H_1^3 [2]$ through replication; Generating $H_1^3 [3]$ through $X_1 \leftrightarrow X_3$ then X_1', X_3' ; Generating $H_1^3 [4]$ through $X_1 \leftrightarrow X_3$; Generating $H_1^3 [5]$ through replication; Generating $H_1^3 [6]$ through $X_2 \leftrightarrow X_3$ then X_2', X_3' ; Generating $H_1^3 [7]$ through $X_1 \leftrightarrow X_3$ then X_1', X_3' . Then assemble $H_1^3 [i]$ according to the order of the Hilbert curve in R^1 to generate H_2^3 . And set H_2^3 as the initial element and repeating the above steps, we can generate H_3^3 .

5 Conclusion

By analyzing the properties of the N-dimensional Hilbert curve, we propose a novel algorithm for obtaining the constructing information of an N-dimensional Hilbert curve. It is easy enough to be used in any application. Now, it is our future problem to implement efficient Hilbert mappings.

References

1. G. Peano, “Sur une courbe qui remplit touteune aire plane,” Math. Ann., 1890, vol. 36, pp. 157–160.
2. S. Biswas, One-dimensional B–B polynomial and Hilbert scan for graylevel image coding. Pattern Recognition, 2004, vol. 37, pp. 789 – 800.

3. R. J. Stevens, A. F. Lehar, and F. H. Preston, "Manipulation and presentation of multi-dimensional image data using the peano scan," *IEEE Trans. Pattern Anal. Machine Intell.*, 1983, vol. PAMI-5, pp. 520–526.
4. H. Chen, Y. Chang. Neighbor-finding based on space-filling curves. *Information Systems*, 2005, vol. 30, pp.205–226.
5. M. F. Mokbel, W. G. Aref Irregularity in multi-dimensional space-filling curves with applications in multimedia databases, *Proceedings of the 10th ACM SIGMIS Information and knowledge management*, Oct 2001.
6. D. Hilbert, Über die stetige Abbildung einer Linie auf ein Flächenstück. *Math. Ann.*, 1891, vol. 38, pp. 459-460.
7. B. Moon, H.V. Jagadish, C. Faloutsos, and J. H. Saltz, Analysis of the Clustering Properties of the Hilbert Space-Filling Curve, *IEEE Transactions on knowledge and data engineering*, 2001, vol.13, pp. 124-141.
8. S. Kamata, R. O. Eason, and Y. Bandou, A New Algorithm for N-Dimensional Hilbert Scanning. *IEEE Trans on Image Processing*, 1999, vol.8, No.7.

DMT: A Flexible and Versatile Selectivity Estimation Approach for Graph Query

Jianhua Feng, Qian Qian, Yuguo Liao, Guoliang Li, and Na Ta

Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China
{fengjh, liguoliang}@tsinghua.edu.cn
{qqpeter99, liaoyg03, dan04}@mails.tsinghua.edu.cn

Abstract. Efficient and accurate selectivity estimation in graph-structured data, specifically for complex branching path query, is becoming a challenging and all-important problem for query performance optimization. Precise and flexible statistics summarization about graph-structured data plays a crucial role for graph query selectivity estimation. We propose DMT, Dynamic Markov Table, which is a dynamic graph summarization based on Markov Table by applying flexible combination of 4 Optimized Rules which investigate local forward and backward inclusions. The efficient DMT construction algorithm DMTBuilder and DMT-based statistical methods are introduced for selectivity estimations of various graph queries. Our extensive experiments demonstrate the advantages in accuracy and scalability of DMT by comparing with previously known alternative, as well as the preferred Optimized Rules that would favor different situations.

1 Introduction

Graph is widely used to model complex and schemaless data, ranging from XML, proteins, to chemical compounds. The key problem for many graph-related applications is how to efficiently process graph query and retrieve corresponding sub-graphs. For achieving best query performance in graph-structured data, effective and accurate estimates for selectivities of both simple and complex path queries are crucially needed by determining the optimal query-execution plan.

XML is a kind of directed labeled graph, which is self-describing and cycle-enabled in nature. The underlying labeled graph model of XML consists of element nodes, which can be simple/complex type value node or composite reference node (i.e. id/idref node[2]). For all the query languages on graph, pattern-based query description is a common and essential feature. This kind of query is more complex than SQL in RDBMS because of its capability in graph navigational query, especially for complicated branching and cyclic query pattern. In general, the most commonly used graph query expressions in graph-structured data, such as XML Database, can be divided into two types: *simple path query* and *complex path query*. The former also has three sub-types: *single path query*; *single-value path query*; *multi-value path query*[5]. *Complex path query*, such as branch-structured or cycle-structured query pattern, can be expressed by combining different kinds of *simple path* expressions.

Generally, a path query P of length n can be denoted by $//t_1/t_2/.../t_n$, with each t_i depicting an element node with predicates. The selectivity for P essentially indicates the count of element t_n that can be reached by P 's navigational process.

2 Graph Summarization Using Markov Table

In general, a common graph summarization model can be described as follow: An original graph data $G = (V, E)$ depicts a directed graph structure. In graph summarization $S = (V', E')$, each $v' \in V'$ with $\text{extent}(v') \subseteq V$ is a representation of the nodes in G classified by certain aggregation principle (i.e. 0-bisimilarity), which records the number of nodes in it. Each edge $(u', v') \in E'$ contains all the edges from $\text{extent}(u')$ to $\text{extent}(v')$. Here, $\text{extent}(v')$ indicates the set of nodes in G corresponding to v' in S .

In *Markov Table* [1], selectivity estimations of paths with length m depend only on the selectivity estimations of sub paths with length $m-1$ preceding it. In fact, the process is modeled as a Markov process of order $m-1$, so this approach is called *Markov Table*. It represents an accurate approximation of the structure of the XML data, but it is only used for simple path query in tree-structured data [1]. In our approach, we extend it to complex branching query in graph-structured data with new features added.

Theorem 1. *In summarization graph S , the frequency of edge (u_i, v) , which is one of all the edges leads to v , can be denoted as follows:*

$$\text{Freq}(u_i, v) = \frac{\text{Freq}(u_i)}{\sum_{u_i} \text{Freq}(u_i)} \times \text{Freq}(v) \tag{1}$$

The computation for the frequency of edge (u_i, v) mainly considers two factors. $\frac{\text{Freq}(u_i)}{\sum_{u_i} \text{Freq}(u_i)}$ presents the fraction of edges with start node u_i in all the edges leading to v . $\text{Freq}(v)$ is the frequency for the node v in S . Hence, the product of these two factors is the approximation for frequency of edge (u_i, v) . This idea is also addressed in [1] and [6], which is a statistical method based on uniformity assumption.

3 DMT

Since the computation of frequency is based on uniformity assumption [6], it is only a coarse estimation that can't supply more exact information about correlations and distributions for sub paths. With the limitation of $S(0)$, we need to investigate optimization methods for refining graph summarization. As a basis for optimized rules, we first exploit the local features about edges in graph summarization S . Two definitions are proposed to describe the types of edges in S , which consider the local forward and backward inclusions.

Definition 1. *Forward-Inclusion (FI): For each edge (u, v) in graph summarization S , if u can reach no nodes except v , then the type of (u, v) is FI.*

Definition 2. *Backward-Inclusion (BI):* For each edge (u, v) in graph summarization S , if v can only be reached by u , then the type of (u, v) is BI.

Therefore, all the edges in S can be classified into 4 types: FI, BI, $FI \wedge BI$ and NI. NI denotes the type of $\neg(FI \vee BI)$. With the definition 1 and 2, we draw two important theorems as strong evidences supporting accurate selectivity estimations.

Theorem 2. Given a path $P = (t_1/t_2/.../t_n)$ in graph summarization S , if types of all edges (t_i/t_{i+1}) in P are BI, then $Freq(t_n)$ is an accurate estimation for path P .

Theorem 3. Given a path $P = (t_1/t_2/.../t_n)$ in graph summarization S , if types of all edges (t_i/t_{i+1}) in P are FI, then each node in $extent(t_i)$ ($1 \leq i < n$) has connected path reaching some node in $extent(t_n)$.

As an example for branching query in sample graph data, query $P = /Tsinghua[C_s/C/DR/IR]/D_s/D$ is a complex path query in which sub paths C_s/C , C/DR and DR/IR have types of FI. After applying *Theorem 3*, $Freq(P) = Freq(/Tsinghua[C_s]/D_s/D)$, which is simplified by pruning C_s/C , C/DR and DR/IR . The intrinsic sense of this simplification is based on a fact that the existent probability of sub path $C_s/C/DR/IR$ is a hundred percent which is concluded from *Theorem 3*.

In order to leverage *Theorem 2* and *3* sufficiently, we propose 4 optimized rules with goals to depict the graph summarization containing more BI and FI type edges. Benefited from these optimized rules, our selectivity estimation methods on DMT perform more accurately.

Optimized Rule 1: Given an edge (u, v) with $\neg FI$ type in graph summarization S , let the type of (u, v) be T . The node u in S can be split into two nodes u_1 and u_2 , where $set(u) = set(u_1) \cup set(u_2)$ and $set(u_1) \cap set(u_2) = \emptyset$, then the new types of (u_1, v) and (u_2, v) are $FI \cup T$ and T respectively. Figure 1(a) shows the process when performing Optimized Rule 1.

Optimized Rule 2: Given an edge (u, v) with $\neg BI$ type in graph summarization S , let the type of (u, v) be T . The node v in S can be split into two nodes v_1 and v_2 , where $set(v) = set(v_1) \cup set(v_2)$ and $set(v_1) \cap set(v_2) = \emptyset$, then the new types of (u, v_1) and (u, v_2) are $BI \cup T$ and T respectively. Figure 1(b) also shows the process when performing Optimized Rule 2.

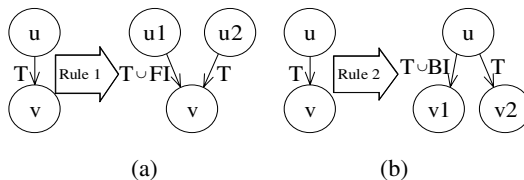


Fig. 1. Process of Optimized Rule 1 & 2

Optimized Rule 3: Given n edges with $\neg FI$ types which start from node u in graph summarization S , let them be denoted as (u, v_1) , (u, v_2) , ..., (u, v_n) . As an assumption, the average frequency of these n edges is a , which is computed by the formula of

$Avg(u, v_i) = \frac{\sum_{v_i} Freq(u, v_i)}{n}$. The node u can be split into two nodes u_1 and u_2 , where $set(u) = set(u_1) \cup set(u_2)$ and $set(u_1) \cap set(u_2) = \emptyset$, such that $Freq(u_1, v_i) > a$ with v_i connected by u_1 and $Freq(u_2, v_j) \leq a$ with v_j connected by u_2 .

Optimized Rule 4: Given n edges with $-BI$ types which start from node u in graph summarization S , let them be denoted as $(u_1, v), (u_2, v), \dots, (u_n, v)$. As an assumption, the average frequency of these n edges is a , which is also computed by the formula of $Avg(u_i, v) = \frac{\sum_{u_i} Freq(u_i, v)}{n}$. The node v can be split into two nodes v_1 and v_2 , where $set(u) = set(u_1) \cup set(u_2)$ and $set(u_1) \cap set(u_2) = \emptyset$, such that $Freq(u_i, v_1) > a$ with u_i leads to v_1 and $Freq(u_j, v_2) \leq a$ with u_j leads to v_2 .

Basing on the Optimized Rules proposed above, we give a concrete demonstration on the DMT construction algorithm: DMTBuilder.

```

Algorithm:   DMTBuilder
Input:      G: Original data graph
            m: Maximum path length supported in DMT
            RuleSet: Optimized Rules Set
Output:     DMT that is optimized by RuleSet
[1]        S(0) := GenS(G); // S(0) contains type of each edge;
[2]        DMT := GenMT(m, S(0));
[3]        For each R in RuleSet do
[4]            DMT := ApplyRule(DMT, R);
[5]        Return DMT;
    
```

Conceptually, this algorithm works as follows:

(1). $S(0)$ is generated with edge types recognition after traversing operation in original data graph G . (2). The first DMT is obtained by $S(0)$ transformation with aggregation principle 0-bisimilarity. (3). Sequentially, each rule in RuleSet is applied on DMT to refine the summarization graph until all rules in RuleSet is executed. RuleSet is a set of Optimized Rules in which one rule can be repeated more than once for certain optimization task.

4 DMT-Based Selectivity Estimation Methods

Simple path query: Theorem 1 in section 3 defines a universal method to calculate the frequency of edge (u, v) . For a given simple path query $P = (t_1/t_2/.../t_n)$, the selectivity of it can be computed by the following formula based on length- m DMT:

$$\mathbf{Freq}(t_1/t_2/.../t_n) = \mathbf{Freq}(t_1/t_2/.../t_m) \times \prod_{i=1}^{n-m} \frac{\mathbf{Freq}(t_{1+i}/t_{2+i}/.../t_{m+i})}{\mathbf{Freq}(t_{1+i}/t_{2+i}/.../t_{m+i-1})} \tag{2}$$

Equation (2) can be inferred from Theorem 1 extending from length 1 to m . The inference process is omitted here because of space. The equation also appeared in [1], but without derivation.

Complex path query: selectivity estimation of *complex path query* is more complicated than *simple path query* because it always involves more than one sub paths. Essentially, only with exact correlation information supplied, accurate estimations can be performed. DMT is a step-by-step optimized graph summarization for selectivity estimation, which has the ability in nature to offer the precision on the statistical information needed in estimations. Given a complex branching path query having two branching sub paths, $P=t_1/t_2/\dots/t_n[t_{n+1}/t_{n+2}/\dots/t_{n+m}]/t_{n+m+1}/t_{n+m+2}/\dots/t_{n+m+k}$, here we assume $P1=t_1/t_2/\dots/t_n$ with length n , $P2=t_{n+1}/t_{n+2}/\dots/t_{n+m}$ with length m and $P3=t_{n+m+1}/t_{n+m+2}/\dots/t_{n+m+k}$ with length k . According to the statistical model for branching query selectivity estimation in [6], following equation is used to compute the selectivity of P .

$$\text{Freq}(P1[P2]/P3) = [r(P1) \times r(P2|P1) \times r(P3|P1)] \times \text{Freq}(t_{n+m+k}) \quad (3)$$

$r(P1)$ denotes occurrence probability of $P1$, as well as posterior beliefs with $r(P2|P1)$ and $r(P3|P1)$. Because of the path independence assumption in [6], we have $r(P2|P1) \approx r(P2)$, $r(P3|P1) \approx r(P3)$.

5 Experiment Study

We use three kinds of standard XML testing datasets in our experiments: Shakespeare[3], XMark and DBLP[4]. We choose 300 simple path queries and 200 complex path queries for each dataset. These queries are obtained by off-line scanning of the graph summarization generated by our $\text{GenS}()$ function in DMTBuilder. AER, an abbreviation for Average Error Rate, is proposed to measure the average value of relative error for different approaches on selectivity estimations. CST[3] does well in trading off the accuracy and memory space, in which Correlated Suffix Trees are used to depict graph summarization. The experimental memory size is from 0 to 50 KB. Optimized Rules 1 to 4 are repeatedly applied till memory size limitation reached. Except combined query set, we also evaluate separately on *simple path query* and *complex path query* to investigate the accuracy of DMT.

Experiment 1: Comparison between DMT and CST.

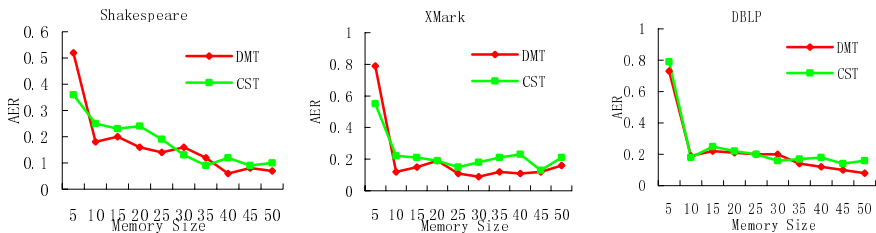
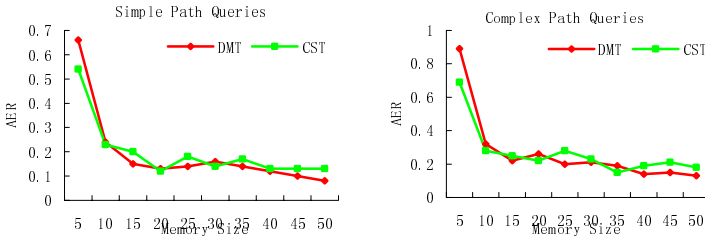


Fig. 2. AER on three datasets

Experiment 2: Experiments on simple path queries and complex path queries.**Fig. 3.** AER on simple and complex path queries

In Figure 2, with the change in memory space constraints, the results of comparisons between DMT and CST show the advantages of DMT in AER. There are some sub conclusions we can draw from our experiments. (1) With the comparisons between DMT and CST on three standard datasets, we have rich proofs to believe that DMT has advantages in accuracy of selectivity estimations. (2) Different usage of Optimized Rules fits for different features of real-life data. From the results above and real data features considered, Optimized Rules 1, 2 are more suitable for graph data with less tags and more long paths. Oppositely, Optimized Rules 3, 4 are good at graph data with more tags and instances for each tag. (3) Figure 3 gives us evidences to believe DMT have more advantages in estimations of complex path queries as more memory space supplied, which is crucial for graph queries.

6 Conclusion

In this paper we presented DMT, a flexible and versatile approach for selectivity estimation of graph query. Our approach has been validated to be efficient and scalable for estimating the selectivities of graph queries, including simple path query as well as complex path query. With the trading off between memory space constraints and accuracy, we exploit the important features of local forward/backward inclusions and propose 4 Optimized Rules for DMT construction. DMTBuilder, as a core algorithm for DMT generation, is fulfilled with dynamic characteristics on Optimized Rules choosing and combination. Our experiments show that DMT performs better for selectivity estimations, especially for complex path queries. This work also can be extended to cyclic graph queries based on DMT's capabilities of summarization and refining.

References

1. A. Abounaga, A. R. Alameldeen, and J. F. Naughton. Estimating the selectivity of XML path expressions for internet scale applications. VLDB 2001.
2. T. Bray, J. Paoli, C.M. Sperberg-McQueen, and E. Maler. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation, October 2000.

3. Zhiyuan Chen, H.V. Jagadish, Flip Korn, Nick Koudas, S. Muthukrishnan, Raymond Ng, and Divesh Srivastava. Counting twig matches in a tree. In Proc. IEEE Int. Conf. on Data Engineering, pages 595–604, Heidelberg, Germany, April 2001.
4. M. Ley. DBLP XML records, 2001.
5. Lim L., Wang M., Padmanabhan S., Vitter J., Parr R.: XPathLearner: An On-Line Self-Tuning Markov Histogram for XML Path Selectivity Estimation. VLDB 2002.
6. N. Polyzotis, M. Garofalakis, Statistical Synopses for Graph-Structured XML Databases, SIGMOD 2002.

A New Public Key Certificate Revocation Scheme Based on One-Way Hash Chain[★]

JingFeng Li^{1,2}, YueFei Zhu¹, Heng Pan¹, and DaWei Wei²

¹Institute of Information Engineering, Information Engineering University,
Zhengzhou 450002, China

²Institute of Electronic Technology, Information Engineering University,
Zhengzhou 450004, China

lee_jingfeng@hotmail.com

Abstract. The *Public Key Certificate* (PKC) is an important way to support the secure global WEB applications. The revocation of the PKC, however, remains as one of the most costly problems in the certificate management. In this paper, a new public key certificate revocation scheme based on one-way hash chain is proposed. Specifically, no centralized authority holds responsibility to collect and publish certificate status information. Instead, the individual user takes charge of his own certificate status by periodically releasing an updated secure hash value. The paper concludes with an informal examination of the security, efficiency and scalability of this scheme.

1 Introduction

To avoid attacks against our WEB applications, we need to provide them with the basic security services such as authentication, access control, confidentiality, information integrity and non-repudiation. The *Public Key Infrastructures* (PKIs) [1] have been deployed to provide these services. Actually, Examples include security protocols such SSL (used in web browsers), IPSec (used in firewalls and desktop computers), S/MIME (a secure e-mail protocol), and SET (the electronic commerce credit card transaction protocol).

The *Certificate Authority* (CA) in PKI issues a *public key certificate* (PKC) that binds the owner's public key to his identity. After a PKC has been created and circulated, however, due to various reasons, it may be necessary revoked prior to the expiration date. Practically, efficient revocation of the public key certificates remains as a difficult issue in PKIs. Till now, there are two well-known certificate revocation schemes: *Certificate Revocation Lists* (CRLs) [2] and *Online Certificate Status Protocol* (OCSP) [3]. Unfortunately, each of them has some drawbacks, which have limited the growth and the practical use of PKI.

[★] This work has been supported in part by the National Natural Science Foundation of China (90204015, 60473021), the Basic Research Program of China (973 Program) (G1999035804) and the Natural Science Foundation of Henan Province (511010900).

In this paper, we propose a new efficient certification revocation scheme based on one-way hash chain. The rest of the paper is structured as follows: In Section 2, the proposed new scheme is described in detail. In Section 3, we summarize the important properties of this scheme in view of security, efficiency and scalability. A conclusion is drawn in Section 4.

2 The New Scheme

2.1 OWHF and Hash Chain

A *one-way hash function* (OWHF) H can map an input M of the arbitrary length to an output of the fixed length, which is called hash value $h : h = H(M)$, where the length of M is m -bits.

OWHF H has the following properties[4]:

- Given a hash value h , it is computationally infeasible to find the input M such that $H(M) = h$.
- Given an input M , it is computationally infeasible to find a second input $M' \neq M$ such that $H(M) = H(M')$.

Till now, the *SHA-1* algorithm and *SHA-2* algorithm are the most widely used OWHFs [5].

Definition 1: Let H be an OWHF and let r be a random number. Thus, a hash chain can be deduced by iteratively hashing r , which can be written as:

$$H^i(r) = H(H^{i-1}(r)), (i = 1, 2, \dots)$$

Where, r is regarded as “*trust anchor*” of the one-way hash chain. The hash chain includes a sequence of hash values, which can be denoted by $h_1 = H(r)$, $h_2 = H(h_1), \dots, h_i = H(h_{i-1})$, $(i = 1, 2, \dots)$.

2.2 Working Principles

The basic idea of our scheme is to divide the lifetime of *public key certificate* (PKC) into short intervals. The individual certificate entity generates a one-way hash chain and keeps the “*trust anchor*” confidential. Each hash-chained element is related to an interval. To refresh the validity of the PKC, the corresponding hash-chained element is updated by the entity at the beginning of each interval. If the “*trust anchor*” and the remaining hash chain are destroyed by himself, the certificate entity will revoke his PKC at the end of current interval. For clarity, the working mechanism is illustrated in Figure.1.

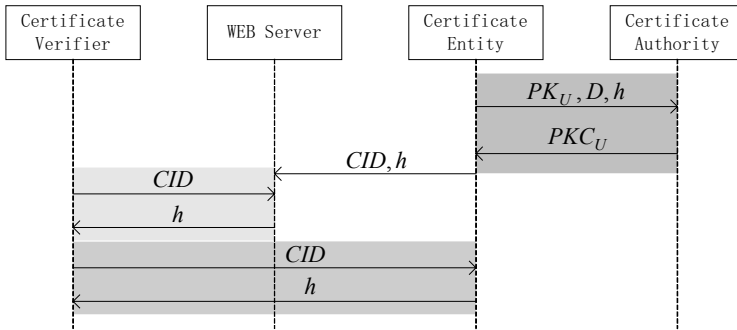


Fig. 1. The Working Mechanism

Our revocation is delineated into three major processes: the PKC generation, the PKC status configuration and the PKC status verification.

2.2.1 Generation of the PKC

Suppose all system clocks are synchronized. The lifetime T of every PKC is the same, so is the updating interval L . where, $T / L = j$ ($j > 1$).

The PKC is generated as follows:

- The entity U generates a pair of keys: public key PK_U and private key SK_U .
- The entity U defines the starting valid date as D . The updating point for the i th interval $[D_{i-1}, D_i)$ is denoted by $D_i : D_i = D + i * L, (i = 1, 2, \dots, j ; D_0 = D)$.
- The entity U selects a random number r , and generates the corresponding one-way hash chain by hashing rj times. The secret r and hash chain are solely dominated by U .
- The entity U sends a PKC request message containing PK_U, D and h_j to CA.
- The CA authenticates this PKC request.
- If the authentication success, the CA will issue a PKC to U , which is denoted by $PKC_U : PKC_U = SIGN_{CA}(CID, U, PK_U, D, h_j)$, where CID represents the certificate ID.

2.2.2 Configuration the PKC Status

To confirm validity of PKC_U in the i th interval, the entity starts to release the hash value h_{j-i} at D_{i-1} and persists it unchanged till D_i (see Fig.2 (a)). If PKC_U should be revoked after the i th interval for some reasons, the entity can stop releasing h_{j-i-1} at D_i and erase the secret r and the remaining hash chain (see Fig.2 (b)). In some scenarios, the entity wants to temporarily suspend PKC_U for m intervals, e.g. the time period from D_i to D_{i+m} . Therefore, he stops releasing hash values ($h_{j-i-1}, h_{j-i-2}, \dots, h_{j-i-m}$) from the updating point D_i , and restarts to release $h_{j-i-m-1}$ at D_{i+m} (see Fig.2 (c)).

The hash values can be released through a public WEB server. That means the hash chain will be sent to the WEB server by individual entity. Note that every entity must inform the WEB server that h_1 should be securely eliminated at D_j . On the other hand, the entity can also directly respond to certificate status queries from the certificate verifiers. Either by secure email or by mobile SMS service, the appropriate hash values can be released.

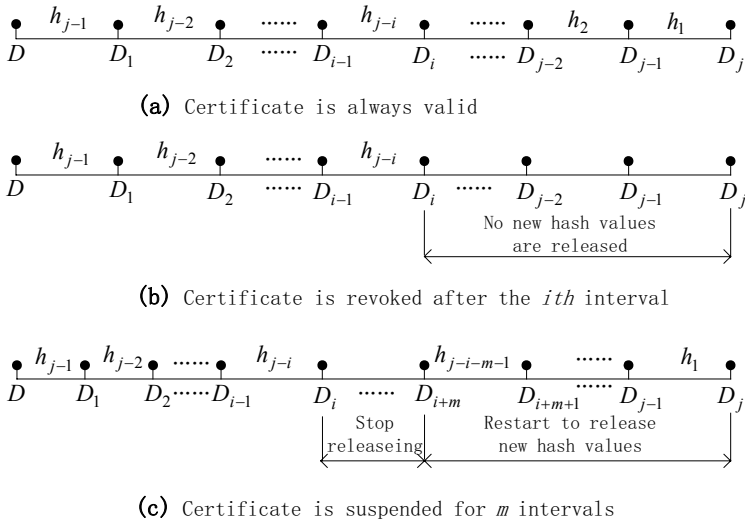


Fig. 2. Certificate Status Update

2.2.3 Verification of the Status

Suppose the current time is T' . When a certificate verifier wants to verify the validity of PKC_U , he can take the following steps:

- The verifier sends a query including CID to the WEB server (or the corresponding certificate entity) and retrieves a hash value from it.
- The verifier extracts the starting valid date D and the hash value h_j from PKC_U .
- The verifier computes the parameter $i : i = \lfloor (T' - D) / L \rfloor$ according to the parameter T' .
- The verifier calculates the value h_{END} by hashing h_{j-i} i times.
- The verifier checks whether h_{END} is equal to h_1 . If true, he believes PKC_U is valid until the next updating point. Otherwise, he affirms PKC_U is already invalid.

3 Performance Analysis

3.1 Security Analysis

A certificate revocation scheme must guarantee integrity and confidentiality of the certificate status information. Our scheme relies on cryptographic primitive OWHF to satisfy these security requirements. Each entity generates and solely controls his secret r and the corresponding one-way hash chain, which will be used to confirm validity of his certificate. If a certificate verifier determines a given hash value is true, he will affirm that this hash value is released by an authentic entity.

To impersonate the owner of PKC_U during the i th ($i = 1, 2, \dots, j-1$) interval, an adversary should release the appropriate hash values h_{j-k} ($k \leq i-1$). He may use only two approaches to achieve it. First, he may get the preimage of h_{j-k} . However, according to the properties of OWHF (see section 2.1), it's computationally infeasible. Second, he may forge the rest values in the hash chain. Once receiving a new hash value, the verifier will generate h_{END} by hashing this hash value several times. Then, he will check whether h_{END} is equal to h_j . Therefore, he can detect any fraud attempt.

Assume the adversary occasionally penetrates a certificate entity. However, when the certificate entity revoked his certificate, he has already erased r and the remaining hash-chain values. Thus, the adversary still can't "resurrect" any previously revoked PKC.

Remarks: The secret r plays a significant role in guaranteeing security of our scheme. So long as r is secure, even if SK_U is compromised, the certificate entity still could revoke his PKC instantly. Strictly speaking, r and SK_U should be stored separately. For example, r can be stored into an *USB-Key*.

3.2 Efficiency Analysis

In our scheme, only j hashing operations need to be done. Obviously, Calculating a hash value is significantly quicker than verifying a digital signature. Thus, compared with the traditional schemes (CRLs or OCSP), the new scheme will save much computation cost. Our scheme shows certificate status information by releasing appropriate hash value, whose maximum length is 32 bytes. Apparently, the required bandwidth is much lower.

As previously noted (Section 2.2.1), h_j is defined and added into the certificate extension field in our scheme. Here, h_j is 20 bytes for SHA-1 algorithm or 32 bytes for SHA-2 algorithm. Since the traditional PKC already consists of a signature (at least 256 bytes long), a public key PK_U (at least 128 bytes long) and other data, the additional hash value is negligible storage cost.

3.3 Flexibility Analysis

Typically, the traditional schemes use the centralized authority (e.g., CA or OCSP Responder) to update certificate status information. However, for the certificate veri-

fier, it's a poor way to obtain single certificate's status, because communication load and computation load are high. Additionally, such centralized entity is very vulnerable to single point failure. Instead of using such a centralized authority, the individual certificate entity that controls on validity of his certificate. Obviously, such a distributed revocation scheme is more flexible and robust than the traditional ones.

4 Conclusions

The PKC is a powerful tool for security-concerned WEB applications. In this paper, we propose a PKC revocation scheme based on cryptographic primitive OWHF. In this scheme, a secret parameter r and the corresponding one-way hash chain are key components, which enable individual entity to take charge of his certificate status.

References

1. ITU-T: Information technology - Open systems interconnection- The directory: Public-key and attribute certificate frameworks. ITU-T Recommendation X.509 (V4). (2000).
2. R.Housley, W.Ford, W.Polk, and D.Solo. : Internet X.509 public key infrastructure certificate and CRL profile. RFC2459. (January 1999).
3. M.Myers, R. Ankney, A. Malpani, S. Galperin and C. Adams. : X.509 internet public key infrastructure online certificate status protocol (OCSP). RFC2560. (June 1999).
4. Bruce Schneier. : Applied cryptography second edition: protocols, algorithms, and source code in C. China Machine Press. Beijing. (2000).
5. National Institute of Standard and Technology. : Secure Hash Standard. FIPS PUB180-2. (2002).
6. Peifang Zheng. : Tradeoffs in certificate revocation schemes. ACM SIGCOMM Computer Communications Review, 33(2). (2003) 103-112

Interactive Chinese Search Results Clustering for Personalization

Wei Liu¹, Gui-Rong Xue¹, Shen Huang¹, and Yong Yu²

¹ Shanghai Jiao Tong University No.800, Dongchuan Road,
Shanghai, China 200240

{liuweimei, grxue, huangshen}@sjtu.edu.cn

² Shanghai Jiao Tong University Computer Science Department,
Shanghai, China 200030
yyu@cs.sjtu.edu.cn

Abstract. Searching for information on the Web has attracted great attention in many research communities. Results returned by most Chinese web search engines usually reach up to thousands or even millions of documents, so efficient interfaces for search and navigation are of critical need. In this paper, we proposed an interactive search results clustering system to facilitate browsing Chinese web pages in a more compact and thematic form. Users could select the clusters that best matched the implicit meanings of their queries and personalized on-the-fly those search results. Our experiments showed that this highly efficient approach outperformed the traditional Chinese search engines.

1 Introduction

With the exponential growth of the Internet in China, it has become more difficult in finding the exactly relevant information. Existing Chinese search engines such as Sina, Sohu and Baidu often return a flat list of search results, given a certain sequence of query words. When the retrieved documents are in thousands or even millions, users have to sift through pages of lists to locate their interested topics. This is time-consuming and the browsing style seems to be unattractive. Given a query, the same results will be produced for different users. Therefore it is necessary to find a solution helping web users focus on their desired topics.

Many English IR-tools have been designed for better results organization and user interaction. Vivisimo [12] and IBoogie [13] added a hierarchy of folders to the flat list of relevant documents, which could be expanded or collapsed. However, when [13] was applied to Chinese Web pages, mess codes often occurred. Others like Mooter [14], Kartoo [15] and Grokker [16] etc. reorganized the search results in different classes and offered good visualization. The faults lay in their lack of efficiency and no support for Chinese search tasks.

In this paper, we proposed an interactive Chinese search results clustering system as a solution and the query processes could be dynamically improved according to users' preferences. Three features for a phrase were calculated and combined to produce a score. Then the ranked phrase list was presented as the name list of candidate clusters. We also offered a good visualization of search interface for user's interaction

and personalization. Users could select clusters that best matched the implicit meanings of their queries, or do the query refinement. Moreover, those search results could be personalized on-the-fly by filtering out the clusters in which users had no interest.

The rest of the paper is organized as follows. Some related work is introduced in Section 2. In Section 3, we described the whole system, including four parts. In Section 4, the experiment results are given. Finally, we draw the conclusions in Section 5.

2 Related Work

Some previous work has been done on organizing the search results into groups based on their overall similarity to one another. Zamir et al. [1] grouped web search results using STC (Suffix Tree Clustering) algorithm and Hearst et al. [2] used the scatter/gather techniques. In [5], Tolerance Rough Set theory has been brought in. Concise, intelligible cluster labels were derived from tolerance classes using special heuristic rules. However, it still seemed to suffer from duplicated labels. [6] introduced a hierarchical monothetic clustering algorithm to build a topic hierarchy for the collection of search results. [7] reformalized the search result clustering problem as a supervised salient phrase ranking problem. These approaches are mainly well suited for English retrieval systems.

As for some other non-English search results clustering methods, [4] indicated that the characteristics of produced clusters in Polish strongly depended on the pre-processing phase. Semantic Hierarchical Online Clustering (SHOC) algorithm [3] combined key phrase discovery and orthogonal clustering to generate clusters. This algorithm can also be applied to the Chinese tasks, but its system is not available for comparisons now.

3 Our Search Results Clustering System

3.1 System Overview

The Interactive Chinese search results Clustering (ICC) System mainly consists of four parts—search results fetching, Chinese segmentation, ranking and clustering engine, and the user interface. The whole architecture is described as follows in Fig.1.

3.2 Multi-thread Crawling and Parsing

The web pages of search results are fetched from one of Yahoo.CN, Sina, Sohu, and Baidu search engines using a multi-thread crawler. Then the pages are immediately sent to text processing pools so that the both the crawling and the HTML parsing can be operated synchronously. The web pages are analyzed by multi-thread HTML parsers as well. For efficiency, the original web pages are not downloaded.

3.3 Chinese Segmentation

Different from the English, Chinese language is lack of explicit separators, i.e. blanks or delimiters, in written oriental sentences to indicate word boundaries. Therefore, we

take advantage of the n-gram and ICTCLAS system[8] for Chinese lexical analysis. ICTCLAS uses an approach based on multi-layer HMM(Hidden Markov Model), which includes word segmentation, Part-Of-Speech tagging and unknown words recognition.

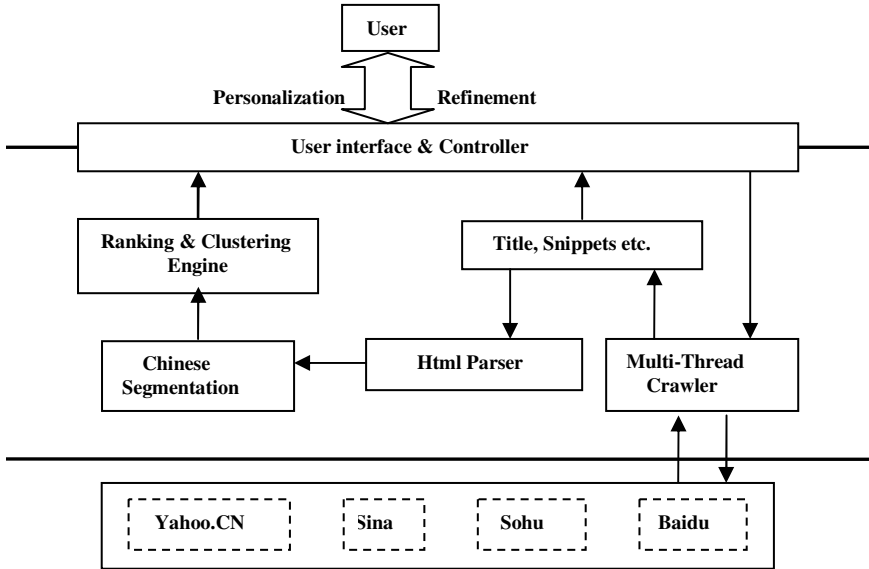


Fig. 1. The Architecture of the Interactive Chinese Search Result Clustering System

3.4 Ranking and Clustering

After segmentation, a linear combination of three features –TFIDF[9], phrase length and cluster compactness¹ is used for each phrase’s ranking:

$$Score = \alpha_1 Tfidf + \alpha_2 Len + \alpha_3 ClusterCompactness \tag{1}$$

where the coefficients α_1, α_2 and α_3 are 0.15, 0.24, -0.02 respectively.

The ranked phrases are kept in descending order as candidate labels and the documents with the key phrases occurring will form the corresponding clusters. However, to eliminate duplicate clusters, Levenshtein Distance(LD) algorithm [10] and Mutual Information (MI) [11] are still needed.

3.5 Interactive User Interface for Personalization

Based on the ranked results, there are eight clusters presented on each page. An example of the first top-8 clusters for the query “手机” (“Mobile phone”) is in Fig.2.

¹ Cluster compactness is the average distance for each document to the cluster centre.

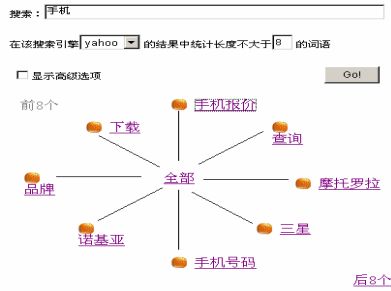


Fig. 2. The Visualized Clusters of Search Results for Query “手机”(“Mobile phone”) on the first page for Yahoo.CN

Users can have a quick view of all the topics in the search results without sifting through the pages of lists. After clicking a cluster button, the documents that belong to that cluster will be listed in the main body of the browser. By selecting a set of labels $L = \{l_1, \dots, l_f\}$ and clicking the “过滤” (“Filter”) button at the bottom of the cluster list, the snippets which do not belong to the folders tagged by L 's labels will be filtered out. As a result, the search results are re-ranked according to user's choice.

If a user is not satisfied with the current search results, he can make use of the “refine” button in **Fig.3**. It adds the selected cluster label l_i to the original query Q , and performs another new search, with $Q' = Q \wedge l_i$ to get more specific results.

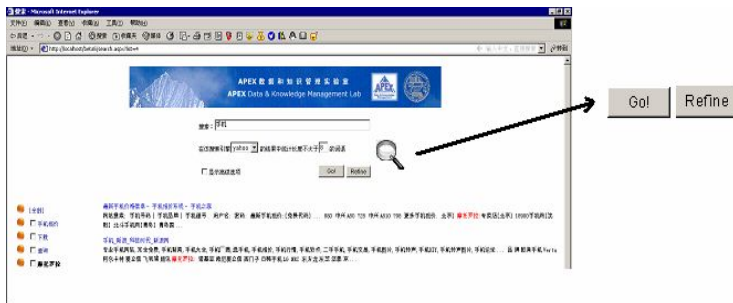


Fig. 3. The “Go” and “Refine” buttons in the browser for query submission and refinement

The system does not require any explicit login by the user nor a pre-compilation of the user profile. But it can help users quickly focus on their interested topics, re-rank the search results and do the query refinement. All of the work is done on the default 100 snippets returned by the queried engine and user can customize it in the “Search Options”.

4 Experiment Results

The Interactive Chinese search result Clustering (ICC) System ran on a PC with P4 2.4GHZ, RAM 1GB. We selected “手机”(“Mobile Phone”) as a query and tested the

average running time. It has been found that Sina and Sohu could finish in 2 seconds for 200 snippets; Yahoo.CN took the second place, within 4 seconds; and Baidu was the slowest, more than 20 seconds. The time spent on search results fetching accounted for 55%~84% of the whole process.

We conducted user studies on the four search engines and got the average number of clusters with good labels($Aver_{gl_c}$) on each page as in Table 1.

Table 1. The average percentage of clusters with good labels in each page

Page	$Aver_{gl_c}$	Percentage
1	6.72	84%
2	5.26	65.75%
3	3.8	47.5%

Then given 26 popular queries, users were asked to evaluate our interactive clustering search engine and compare it with other traditional Chinese search engines like Yahoo.CN and Baidu etc.

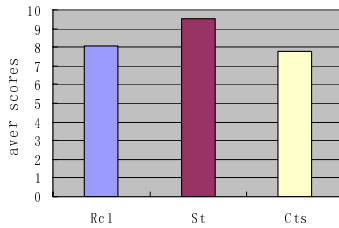


Fig. 4. Users’ evaluation of the system’s whole performances.(“ReI” is the “Representation of cluster labels”, “St” is the “Search time” and “Cts” is the “Comparison with other traditional Chinese search engines”). 10: excellent; 7: comparatively good; 5: ordinary; 2: tolerable;0: bad.

It could be seen that most users thought the time efficiency of our system was satisfying enough; the cluster labels and the interactive user interface were comparatively good or even better.

5 Conclusions

In this paper, we built up an interactive Chinese search result clustering (ICC) system for personalization. The snippet-based ranking and clustering method were proposed to organize the search results into different themes and offer meaningful cluster labels. We also provided a good visualized user interface, which allowed users to filter and re-rank the search results according to their personal choices or do query refinement. With the reasonably fast speed, users could benefit from our systems.

For future work, we expect to explore the hierarchical organization of clusters and more intelligent search results clustering algorithms. The investigation of relationships between the clusters can also be considered.

References

1. Zamir, O., Etzioni, O.: Grouper: A Dynamic Clustering Interface to Web Search Results. In Proceedings of 8th International World Wide Web Conference, Toronto, Canada (1999) 1361 - 1374
2. Hearst, M., Pedersen, P.: Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results. In Proceedings of 19th Annual International ACM/SIGIR Conference, Zurich (1996) 318 - 329
3. Zhang, D., Dong, Y.: Semantic, Hierarchical, Online Clustering of Web Search Results. In Proceedings of the 6th Asia Pacific Web Conference, Hangzhou, China (2004) 69-78
4. Weiss, D., Stefanowski, J.: Web Search Results Clustering in Polish: Experimental Evaluation of Carrot. In Proceedings of Intelligent Information Processing and Web Mining Conference, Zakopane, Poland (2003)
5. Chi Lang, N., Hung Son, N.: A Tolerance Rough Set Approach to Clustering Web Search Results. In Proceedings the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, (2004) 515-517
6. Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., Krishnapuram, R.: A Hierarchical Monothetic Document Clustering Algorithm for Summarization and Browsing Search Results. In Proceedings of 13th International World Wide Web Conference, (2004) 658-665
7. Hua-jun, Z., Qi-Cai, H., Zheng, C., Wei-Ying, M., Jinwen, M.: Learning to Cluster Web Search Results. In Proceedings of 27th Annual International ACM SIGIR Conference (2004)
8. Huaping, Z., Hongkui, Y., Deyi, X. and Qun, L.: HHMM-based Chinese Lexical Analyzer ICTCLAS. In the Second SIGHAN workshop affiliated with 41th ACL, Sapporo Japan. (2003)
9. Salton, G. Developments in automatic text retrieval. *Science*, 253:974-979.(1991)
10. V. I. Levenshtein: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Cybernetics and Control Theory* 10 (1966) 707-710.
11. T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.
12. <http://vivisimo.com/>
13. <http://www.iboogie.com/>
14. <http://www.mooter.com/>
15. <http://www.kartoo.com/>
16. <http://www.groxis.com/service/grok>

Efficient Delay Aware Peer-to-Peer Overlay Network

Da-lu Zhang and Chen Lin

Department of Computer Science and Technology,
Tongji University, Shanghai 200092, China
daluz@ieee.org
spalding@sohu.com

Abstract. In many P2P systems, the path between any two nodes is mainly decided by their topology of overlay network instead of physical one. However, peers are scattered in distributed geography, requests may route across different Autonomous Systems (AS). So, the inconsistency between overlay network and physical one can not ensure the routing efficiency in the real world. We introduce DAPS (Delay Aware P2P System), a new P2P overlay network that considers both overlay and physical networks. DAPS adopt end-to-end delay as the performance metrics and use “pruning flooding” to achieve efficient routing.

1 Introduction

Peer-to-Peer (P2P) networks have become the fast growing and the most popular Internet application in recent years. Traditionally, P2P network can be roughly divided into two aspects according to its organization: unstructured and structured.

Whether being an unstructured or structured network, there is a same point. Peers in P2P network communicate in a logical overlay network. In this way, the path between any two nodes is mostly decided by the hops in the overlay network level, namely the logical distance between two nodes. But, in fact, the routing efficiency of P2P overlay network largely depends on the end-to-end delay in network level, namely the physical distance. Because all nodes are scattered in distributed geography, some requests may route across different Autonomous System (AS), while other requests only hop in the same network. That the actual topology of network is inconsistent with overlay network may lead to a problem: the shortest path in overlay network level can not ensure the efficiency in real world.

We propose a new P2P system model DAPS that consider both overlay and physical networks, and use end-to-end delay as the performance metrics to achieve efficient routing.

The rest of the paper is organized as follows. Section 2 introduces and describes the related work on P2P systems. Section 3 introduces the concepts and design behind DAPS. Section 4 presents early results from our simulation environment and some conclusions are drawn in Section 5.

2 Related Work

There exist a series of scalable overlay networks, such as Pastry [1], Plaxton [2], Tapestry [3], Chord [4] and Can [5], all offering DHT service. Their common theme

is that they arrange keys (such as lookup items, files, services, etc) and peer nodes in the same identifier space.

Although all these P2P networks work efficiently in the overlay network level, they neglect the real geographic layout more or less. So to some extent, these overlay networks are not really efficient in practice. In general, P2P network is a kind of overlay networks built on existing physical network. Thus, the structure of overlay and physical topology both affect the performance of the whole P2P network. Some P2P overlay network mentioned above doesn't care much about the topology of network.

Some researches [6] have revealed the importance of the topology of physical network, and propose topology-aware overlay network. In order to achieve good performance, the topology-aware overlay network care both the organization of the overlay network and consider some factors of the physical network.

Geographic layout [8] was explored as one topology-aware technique to improve the routing performance in CAN. The technique attempts to map the d-dimensional space onto the physical network such that nodes that are neighbors in the d dimensional space are close in the physical network. This technique can achieve good performance but has the disadvantage that it is not fully self-organizing; it requires a set of well-known landmark servers. In addition, it may cause significant imbalances in the distribution of nodes in the CAN space, leading to hot-spots.

Proximity routing [9] is another kind of topology aware routing. With proximity routing, the overlay is constructed without regard for the physical network topology. The technique exploits the fact that when a message is routed, there are potentially several possible next hop neighbors that are closer to the message's key in the id space. The idea is to select, among the possible next hops, the one that is closest in the physical network or one that represents a good compromise between progress in the id space and proximity. But it will increase the overhead of node maintenance and the size of routing tables. Proximity routing offers some improvement in routing performance, but this improvement is limited by the fact that a small number of nodes sampled from specific portions of the node ID space are not likely to be among the nodes that are closest in the network topology.

Miguel Castro [7] presents a kind of proximity neighbor selection to build structured P2P overlay network with topology-aware routing. Its routing algorithm makes the identifier in routing table point to the close node in physical network. Proximity neighbor selection can improve the performance of the P2P network based on matching, such as Tapestry and Pastry. In Tapestry and Pastry, a message is normally forwarded in each routing step to a nearby node, according to the proximity metric, among all nodes whose node ID shares a longer prefix with the key. Moreover, the expected distance traveled in each consecutive routing step increases exponentially, because the density of nodes decreases exponentially with the length of the prefix match. The routing algorithms in Pastry and Tapestry claim that they allow effective proximity neighbor selection because there is freedom to choose nearby routing table entries from among a large set of nodes.

3 DAPS DESIGN

3.1 Overview of DAPS

DAPS (Delay Aware P2P System) is not a comprehensive peer-to-peer system, but a kind of solution to the problem of topology-aware routing. Its goal is to reduce the time of L for a lookup request and improve the total performance of the P2P system. The main idea is that routing table is divided into several sectors according to delay from low to high, and source node will define a delay boundary, namely, the pruning factor, L_t . Request messages will only send to the nodes whose delay is not more than L_t . Compared with traditional flooding, the request message is largely reduced and realize “pruning flood” with the help of its routing table.

The overlay network of DAPS organizes loosely and sends routing messages with being aware of the conditions of the physical network. Considered the factors affect the total performance, “pruning flood” can lower the network traffic, reduce the complexity of locating algorithm and find expected results efficiently.

With clustered entry in the routing table and the loose organization, the overlay network of DAPS is between structured and unstructured. So, it supports partial-match query and nearly does not need to care much about when the nodes join, leave or fail in the network. And its routing table ensures the adoption to the change of network.

3.2 Routing Table

Each DAPS node maintains a routing table which indicated where to find the destination of lookup request. The structure of routing table is organized on the base of delay, which is divided into clusters from low to high. Nodes in the same clusters have the same range of delay.

If the range of delay is divided into $\sigma_1, \sigma_2 \dots \sigma_i$, and $0 < \sigma_1 < \sigma_2 < \dots < \sigma_i$. The n th row of the routing table contains the nodes whose delay is between σ_{n-1} and σ_n, \dots . In current research, the delay range is separated in same interval, $\sigma_i = i \times \sigma_1$. Each entry records the IP address of the node. So, the routing table arranges its neighbor nodes into several sectors according the delay between them.

3.3 Locating and Routing

In some unstructured P2P system, flooding is a simple solution to route request messages. The search will be iterative until the request is satisfied, or the maximum depth limit has been reached. Because the number of nodes at each depth grows exponentially, the cost of query will be multiplied. In order to lower the cost and reduce the number of visited nodes, DAPS uses “pruning flood” to route request.

Pruning flood is an iterative deepening, multiple breadth-first search with a pruning boundary. Pruning flood is implemented as follows: first, source node will define a delay boundary, namely, the pruning factor, L_t . It means the request must be satisfied in delay of L_t , if the results can be found. Then, the source node will only send requests to the nodes in the routing table where $\sigma_i \leq L_t$, and then $L_t \leftarrow L_t - \sigma_i$. The

requested node then receives and processes the message. If the node has the expected result, it stops the query immediately and returns results to the source node. Otherwise the node will resend the request to the nodes in its routing table where $\sigma_i \leq L_t$. Just like above, the procedure will repeat until $L_t < \sigma$. Neighbor nodes are organized by the time of delay in the routing table, so the number of visited node can be greatly reduced and locates the destination node in physical network not more than L_t or far less.

Assume that there exist N nodes in the network and each node records N/r nodes in its routing table averagely. r is the scale parameter. It means each node knows $1/r$ nodes in the whole network. The routing table is divided into q sectors according to the delay range. So there are $N/(qr)$ nodes in each sectors in average. If $L_t = k\sigma$, in the worst condition, the number of request message that nodes totally send in one lookup procedure will be $(N/qr) (N/qr+1)^{k-1}$.

And, $M \sim O((N/qr)^k)$, M is the number of request message. In the traditional unstructured P2P system, such as Gnutella, it sends message by flooding. The number of its request messages will be $O(N^t)$, t is the TTL of request messages. As it can be seen, if we appropriately choose the parameters of q, r, k , the message number can be largely reduced, and realize "pruning flood".

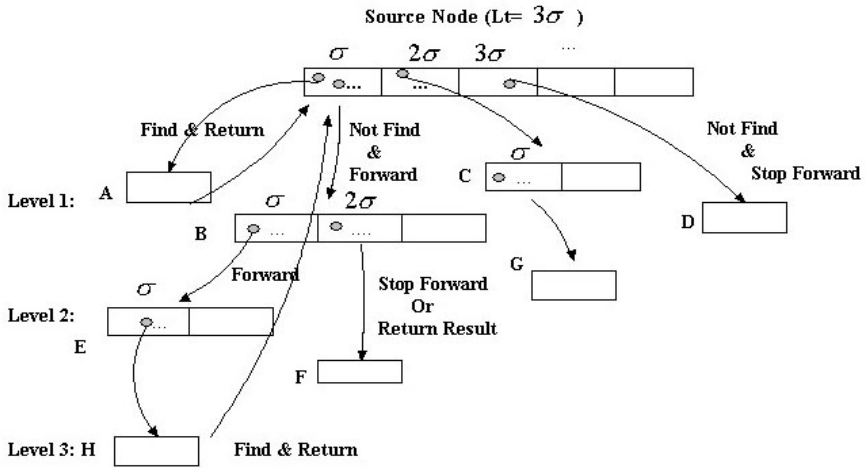


Fig .1. Picture describes the procedure of lookup and routing message with a pruning boundary

The source node starts looking up request with boundary $L_t = 3\sigma$. At first, it only sends request message to nodes whose delay is not more than 3σ . When message reaches node A and finds expected results, it will stop searching and return results at once. Node B receives the request but not find the expected result, so it will forward the request to the node whose delay range is not more than 2σ ($2\sigma = L_t - \sigma$) in its routing table. Again, the request will send to E and F. Not finding result and not exceeding its pruning boundary, E forwards request to H where the result is found and

return it to the source node. As for node F, if there doesn't exist expected results, node F have to exit its request process, as the pruning factor has reached its boundary ($\sigma + 2\sigma = L_t$). Node C, G and D will process respectively.

The algorithm of DAPS is a kind of proximity routing. It adopts end-to-end delay as the performance metrics, as it is more sensitive to user. When searching a request, DAPS select a set of nodes with low delay. It may increase hops in overlay network, but it delivers the lowest delay to user as much as possible.

4 Evaluation

We test the algorithms by simulation on GT-ITM [10] Internet. Figure 2 shows the quality of DAPS routing algorithms. With the increase of pruning factor, the percentage of successful lookup is increasing fast. Nearly about 90% expected results can be found in the delay range (pruning factor) of 800. Further simulation results reveal that we can find nearly all the results with the increase of L_t .

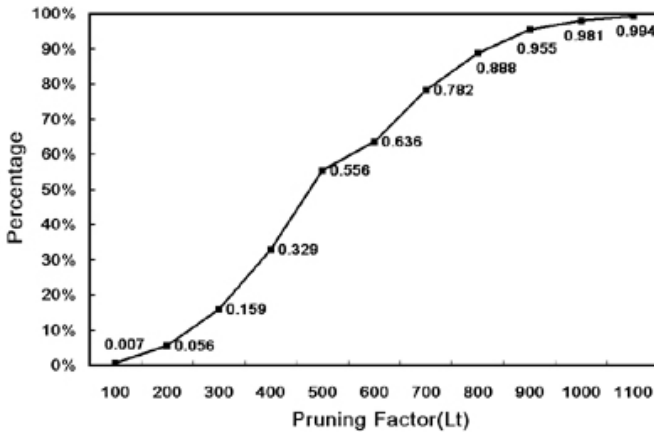


Fig. 2. Picture describes the percentage of successful lookup

5 Conclusion

In most P2P systems, the topology of physical network is often inconsistent with that of overlay network, which may lower the efficiency the routing algorithm. Based on this reason, we propose DAPS which sends request messages by pruning flooding. Compared with some unstructured P2P system, DAPS can efficiently find expected results in the given delay time and the number of request message is largely reduced.

The overlay of DAPS is loosely structured so that the maintenance is much easier and simpler than that of the structured P2P system. DAPS uses pruning flood to send messages, so the number of request messages is still more than that of structured systems which often use DHT to eliminate flooding. So DAPS is not an ideal solution in the large network currently, but the search is more flexible than structured system,

as it supports partial match. In the future work, we will combine pruning flood and DHT into DAPS so as to reduce the request number and improve its routing efficiency further.

References

1. A. Rowstron and P. Druschel, Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. Accepted for Middleware, November 2001.
2. C. Greg Plaxton, Rajmohan Rajaraman, Andréa W. Richa. Accessing nearby copies of replicated objects in a distributed environment. ACM Press New York, NY, USA Pages: 311-320 Series-Proceeding-Articles. 1997. ISBN: 0-89791-890-8.
3. John Kubiawicz, David Bindel, et al. OceanStore: An Architecture for Global-Scale Persistent Storage. In Proceedings of the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), November 2000.
4. Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*, ACM SIGCOMM 2001, San Diego, CA, August 2001.
5. Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, Scott Shenker. *A Scalable Content-Addressable Network*. In Proceedings of the ACM SIGCOMM, 2001.
6. Z. Xu, C. Tang, and Z. Zhang. Building topology-aware overlays using global soft-state. Technical Report HPL-2002-281, HP Labs, September 2002. Submitted for publication, available at <http://www.cs.rochester.edu/u/salTmor>.
7. M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron, Topology aware routing in structured peer-to-peer overlay networks. Tech. Rep. MSR-TR-2002-82, Microsoft Research, One Microsoft Way, Redmond, WA 98052, 2002.
8. S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically aware overlay construction and server selection," in Proceedings of IEEE INFOCOM'02, New York, NY, June 2002.
9. Roberto Rinaldi. Routing and data location in overlay peer-to-peer networks. Diploma thesis, Institut Eurecom and Università degli Studi di Milano, June 2002. Also available as IBM Research Report RZ-3433.
10. E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internet work. In Proceedings of INFOCOM 96, 1996. Scenario Mig Prob New Part Life Part Source

PIES: A Web Information Extraction System Using Ontology and Tag Patterns

Byung-Kwon Park¹, Hyoil Han², and Il-Yeol Song²

¹ Dong-A University, Busan, Korea
bpark@dau.ac.kr

² Drexel University, Philadelphia, PA 19104, USA
hyoil.han@cis.drexel.edu, songiy@drexel.edu

Abstract. We propose a new web information extraction system, PIES, to convert web information into XML documents. PIES uses a user-specified ontology and HTML tag pattern descriptions. The ontology validates the web information the pattern descriptions extract. We designed a new language to describe HTML tag patterns and extraction rules. We implemented PIES and applied it to the US patent web site for evaluation.

1 Introduction

To query web pages, extracting the encoded information in web pages and converting it into structured data (e.g. relational data for SQL) or semistructured data (e.g. XML data for XQuery) is necessary. Now, a number of systems or tools are available for web information extraction [1,4,5,8,9,10]. They can be classified into two categories: automatic or manual. Automatic approaches are possible for the web pages having a predefined schema and encoded with structured data [2,3,6,7].

A lot of web pages, however, include unstructured data as well. For instance, US patent web pages [11] include structured data (e.g. patent number, title, inventors, and registration date) and unstructured data (e.g. claims, drawings, and details). The web pages have no predefined schema, and users should specify an ontology for the data they want to extract. For example, Embley et. al [7] specified an ontology using object-relationship model.

To extract data that match with an ontology, users should provide a guideline for matching the ontology and web pages. In this paper, we propose a new web information extraction system that uses an ontology and HTML tag pattern descriptions as guideline. We implemented the proposed system and applied it to the US patent web site for evaluation.

2 Ontology

To extract information from web pages, the user should first define the ontology about web pages. In this paper, the ontology consists of the conceptual model

of the data the user wants to extract. The conceptual model is defined using a UML class diagram. Figure 1 shows the conceptual model for the US patent data to be extracted. The conceptual model is exported into a textual format – the Petal format – for PIES to process. The Petal format is automatically generated by Rational Rose.

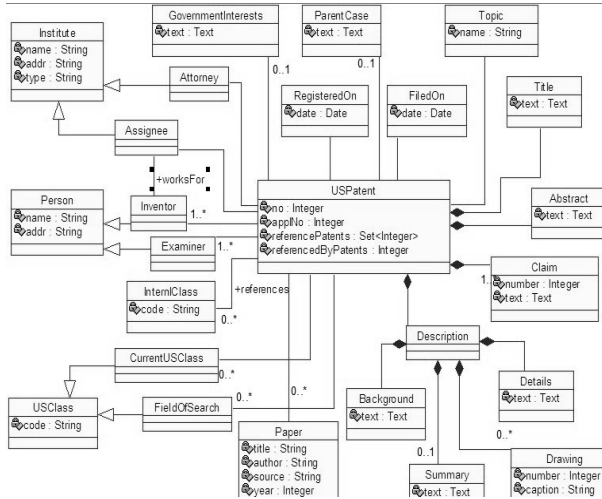


Fig. 1. User-specified ontology for US Patent

3 xRule: Extraction Rule Description Language

xRule is a language with which we can specify the extraction rules for the attribute values of the objects specified in the ontology. An xRule statement is for a single attribute.

Basic Syntax: An xRule statement has such structure as shown in Figure 2. A class name comes first following the keyword 'CONTEXT'. And then, an attribute name comes at the left-hand side of the operator '=' and the TagRegx expression at the right-hand side. The return value of the TagRegx expression is set to the value of the attribute.

Variable: An xRule statement can have a variable. The variable stores an HTML tag sequence matched with the specified TagRegx expression as shown in Figure 3. All variables in xRule are global and used in an iteration statement as explained below.

Iteration: There are two kinds of iteration statements: one is for array type (Figure 4) and the other for multiple instance objects (Figure 5). A nested iteration is also possible (Figure 6).

```
CONTEXT: ClassName
attributeName = TagRex expression for HTML tag sequence
```

Fig. 2. Basic Structure of xRule Statement

```
_variableName = TagRex expression for HTML tag sequence
```

Fig. 3. Variable Definition

```
CONTEXT: ClassName
for _variableName repeat
  attributeName = TagRex Expression
end
```

Fig. 4. Array Type

```
for _variableName repeat
  CONTEXT: ClassName attributeName = TagRex expression
end
```

Fig. 5. Multiple Instance Objects

```
for _variableName repeat
  for _variableName repeat
    CONTEXT: ClassName attributeName = TagRex Expression
  end
end
```

Fig. 6. Nested Iteration

```
set location = TagRex Expression
```

Fig. 7. Set Location

Set Location: The matching always starts from the beginning of a web page. However, we can change the starting location from which the matching begins. Figure 7 shows the syntax of the statement 'Set Location'. The TagRex expression specifies the starting location.

xRule is simple to learn, and powerful enough to provide the functions: arrays, multiple instance objects and nested iteration. It can well match with an object-oriented model. Furthermore, since the rules described in xRule are independent from each other, executing rules in parallel is possible for performance.

4 TagRex: Regular Expression Language for Tag Sequence

TagRex is a specially designed regular expression language to describe the pattern of HTML tag sequences. A TagRex expression is used in an xRule rule. Table 1 shows all the TagRex expressions used in xRule rules. TagRex expressions for HTML tag strings are different from regular expressions for character strings. In a TagRex expression, the basic matching unit is a single HTML tag. We consider a text segment in an HTML document as a single tag. Then, we can consider an HTML document as an HTML tag sequence. A TagRex expression, delimited by the symbol '#', is a pattern with which we search an HTML tag sequence for matching data and returns the matched data. The returned data is set to an attribute value of an object. Both symbols '%' and '~' match with any number of HTML tags. The symbol '%' returns the data matched, while the symbol '~' skips. In some cases, we need to make a choice because web pages use different words for the same thing. We use the symbol '|' for the choice. A web page may contain many text segments. TagRex allows for using a common regular expression, delimited by the symbol '\$', for character string. In some cases, the value we want to search for is contained in an HTML tag's attribute value. The symbol '@' specifies the attribute of an HTML tag.

Table 1. Summary of TagRex expressions

Expression	Description
#...#	delimiter of TagRex expression
#<a>~#	match <a> and any HTML tags including
#<a>~!#	match <a> and any HTML tags excluding
#<a>~!(<c>)#	match <a> and any HTML tags excluding or <c>
#<a>%#	match <a> and any HTML tags including , and return the HTML tags between <a> and
#<a>%!#	match <a> and any HTML tags excluding , and return the HTML tags between <a> and
\$. . \$.	common regular expression for character string
#<a>@href=\$. . \$.#	match the attribute href's value of the tag <a>

With TagRex, we can describe a pattern based on tag level, not on character level. Thus, writing patterns for HTML tag sequences is significantly easy because the number of tags is much smaller than that of characters in an HTML web page. For a US patent web page, the former is more than ten times smaller than the latter.

5 PIES: Pattern-Based Web Information Extraction System

Figure 8 shows the overall architecture of PIES. It consists of two main modules: Rule Matching Engine (RME) and XML Document Generator (XDG). RME

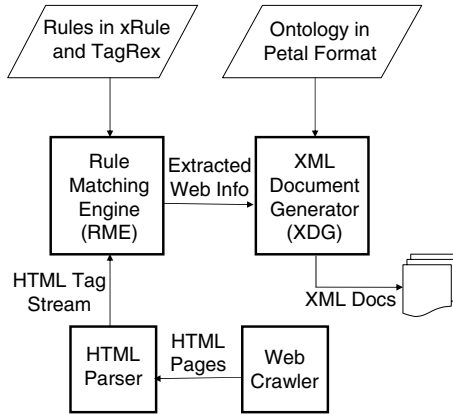


Fig. 8. Overall Architecture of PIES

inputs the rules expressed in xRule from a user and compiles them to efficiently match with web pages. RME also inputs the web pages collected by the web crawler in the form of HTML tag sequence generated by the HTML parser. RME matches the xRule rules with HTML tag sequences.

The function of XDG is to transform the web information RME extracted into XML documents. XDG inputs the ontology expressed in the Petal format and the extracted web information expressed in objects from RME. XDG validates the objects based on the ontology. If the objects are valid, XDG converts them into XML documents. When XDG generates XML documents, it converts an object into an XML element.

PIES has been implemented in the Java programming language on the PC platform with 2GHz CPU and 512MB RAM. We evaluated PIES with respect to performance and robustness. We tested more than thousands of US patent web pages. Even though all the US patents does not have the same structure of web pages, the system successfully extracted all the information specified in the ontology. Furthermore, it extracted and generated a single XML document in less than one second.

6 Conclusions

In this paper, we proposed a new web information extraction system. The main contributions of this paper are as follows: (1) We designed a new language, xRule, to describe the extraction rules for the attribute values of the objects in the user-specified ontology. We deal with the web pages having no predefined schema and containing both structured and unstructured data. (2) We designed a new regular expression language, TagRex, to describe a pattern of HTML tag sequence. (3) We implemented the proposed web information extraction system, PIES, and extracted more than thousands of U.S. patent information from the

real U.S. patent web pages. The U.S. patent has a lot of information. Thus, users need to specify their own ontology about the information they want to extract.

Acknowledgment

This work was supported by the Post-doctoral Fellowship Program of Korea Science & Engineering Foundation (KOSEF).

References

1. B. Adelberg, "NoDoSE - A tool for Semi-Automatically Extracting Structured and Semistructured Data from Text Documents," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, pp. 283–294, Seattle, 1998.
2. A. Arasu and H. Garcia-Molina, "Extracting Structured Data from Web Pages," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, pp. 337–348, San Diego, June 2003.
3. C. Chang and S. Lui, "IEPAD: Information Extraction based on Pattern Discovery," In *Proc. Int'l Conf. on World Wide Web (WWW10)*, pp. 681–688, Hong Kong, May 2001.
4. C. Y. Chung, M. Gertz, and N. Sundaresan, "Reverse Engineering for Web Data: From Visual to Semantic Structures," In *Proc. Int'l Conf. on Data Engineering (ICDE02)*, pp. 363–374, San Jose, California, 2002.
5. V. Crescenzi and G. Mecca, "Grammars Have Exceptions," *Information Systems*, Vol. 23, No. 8, pp. 539–565, 1998.
6. V. Crescenzi, G. Mecca, and P. Merialdo, "RoadRunner: Towards Automatic Data Extraction from Large Web Sites," In *Proc. Int'l Conf. on Very Large Data Bases*, pp. 109–118, Rome, 2001.
7. D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, D. W. Lonsdale, Y.-K. Ng, and R. D. Smith, "Conceptual-model-based data extraction from multiple-record Web pages," *Data & Knowledge Engineering*, Vol. 31, No. 3, pp. 227–251, 1999.
8. J. Hammer, H. Garcia-Molina, S. Nestorov, R. Yerneni, M. Breunig, and V. Vassalos, "Template-Based Wrappers in the TSIMMIS System," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, pp. 532–535, AZ, USA, 1997.
9. A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira, "A Brief Survey of Web Data Extraction Tools," *SIGMOD Record*, Vol. 31, No. 2, pp. 84–93, June 2002.
10. A. Sahuguet and F. Azavant, "Looking at the Web through XML glasses," In *Proc. IFCIS Intl Conf. on Cooperative Information Systems (CoopIS99)*, pp. 148–159, 1999.
11. United States Patent and Trademark Office, <http://www.uspto.gov/>

An Algebraic Framework for Schema Matching

Zhi Zhang¹, Haoyang Che², Pengfei Shi¹, Yong Sun³, and Jun Gu³

¹ Institute of Image Processing and Pattern Recognition, Shanghai Jiaotong University,
Shanghai 200030, China
{zzh, pfshi}@sjtu.edu.cn

² Institute of Software, The Chinese Academy of Sciences,
Beijing 100080, China

³ Department of Computer Science, Science & Technology University of Hong Kong
Hong Kong, China

Abstract. It is well known that a formal framework for the schema matching problem (SMP) is important because it facilitates the building of algorithm model and the evaluation of algorithms. First, based on universal algebra, we propose a meta-meta structure for schema. This definition has a distinctive feature: it is able to describe any particular style of schemas, and transforms a schema and other matching information into a finite structure over specific signature. Then, we formalize SMP as a schema homomorphism problem, and prove that SMP is equivalent to finding a semantic homomorphism from one schema to another. These results lead to the main contribution of this paper: an algebraic framework for SMP. Thirdly, we show a classification of schema matching based on the algebraic framework.

1 Introduction

The schema matching problem (SMP) plays a key role in various database applications [13]. The goal of schema matching is to find the semantic correspondences between elements of two schemas. For SMP, a major challenge is in properly addressing semantics: the semantic representations associated with the schemas. There are many practical techniques for representing the semantics of schemas, such as semantic data models, description logics, and corpora or dictionaries, etc. A second major challenge is in developing such a framework that is applicable to a variety of data models, such as the relational, object-oriented, and XML models. The framework facilitates the building of algorithm model and the evaluation of algorithms.

To achieve semi-automatic schema matching, there are many methods have been proposed. However, these practical matching approaches have not utilized the theoretical foundation and most of them have relied on ad-hoc approaches. Doan *et al.* [3] developed a LSD system that uses machine-learning techniques (Bayesian learners) to match a pair of schemas; Madhavan *et al.* [8] implemented a CUPID system, which uses similar-based heuristics algorithms to achieve semi-automatic schema matching. Melnik [9] carried out a generic model-management tool – RONDO, and proposed a graph-based algorithm - similarity flooding for schema matching. For a recent review of SMP, we can refer to a survey [13].

The theoretical foundation and the formal framework for related problems of SMP have been developed. Paolini and Pelagatti [12] analyzed mappings between external views of a database and conceptual views of the database itself, where database was represented by many-sorted algebra and mappings were treated as homomorphisms; Hull [7] presented a tutorial for describing fundamental problems that are raised by semantic heterogeneity and surveyed theoretical frameworks that can provide solutions for them. Most work focused on schema integration in federated databases; For solving schema equivalence, based on a graph model (*Schema Intension Graph*), Miller *et al.* [10] proposed that the isomorphic schemas are equivalent, and discussed the structural transformation of schemas; Alagić and Bernstein [1] developed a categorical model theory for generic schema management. By using the category to represent the schema, schema morphisms are defined as mappings of schema signatures that preserve the integrity constraints. Schema transformations within a particular category of schemas are viewed as morphisms of that category. Then, they built the formal frameworks for schema integration and transformation. However, little attention has been given to the theoretical foundation of SMP. Therefore, in this paper, we address to develop a formal framework for SMP based on universal algebra.

The rest of this paper is organized as follows. Section 2 proposes a formal matching-oriented definition of schema, which is based on universal algebra. Section 3 focuses on individual matching, and introduces the formal definition of schema matching. Further, section 4 studies schema homomorphism. We prove that SMP is equivalent to finding a semantic homomorphism from one schema to another. Then, section 5 presents a new taxonomy for SMP under the algebraic framework. Section 6 summarizes the contributions and suggests future research directions.

2 Schema

To build the algebraic framework, we present a formal *meta-meta structure (model)* to describe the various schemas at first. Schemas are *finite structures* over the specific signatures. The matching objects and their properties are able to assemble in the structure. By the basic definition of structure [4, 5], we define a *generic matching-oriented schema*, which is based on universal algebra.

Definition 1. (*Schema*) A schema \mathcal{S} is a finite structure over a signature σ , consists of individual set $I^{\mathcal{S}}$, label collection $Lab^{\mathcal{S}}$, function set $F^{\mathcal{S}}$, relation set $R^{\mathcal{S}}$, written a 4-tuples $\mathcal{S} = (I^{\mathcal{S}}, Lab^{\mathcal{S}}, F^{\mathcal{S}}, R^{\mathcal{S}})$, where,

1. σ is a finite collection that is composed of individual symbols, label symbols, function symbols, and relation symbols, where, each function symbol f or relation symbol R , respectively comes associated with an arity, $ar(f)$ and $ar(R)$, which are non-negative integers.
2. $I^{\mathcal{S}} = \{s_1, s_2, \dots, s_n\}$ is a finite nonempty set that includes individuals, which denote the prepared-matching objects. Each of them is uniquely identified by an object identifier (OID).
3. $Lab^{\mathcal{S}} = \{Lab_1^{\mathcal{S}}, Lab_2^{\mathcal{S}}, \dots, Lab_i^{\mathcal{S}}\}$ is a finite constant collection that includes the label sets for individuals. The labels are the strings for describing the properties of individuals.

4. $F^S = \{f_1^S, f_2^S, \dots, f_j^S\}$ is a finite set that includes the labeling functions, which are partial function. The domain of each function is the individual set, accordingly, the codomain is the label collection.
5. $R^S = \{R_1, R_2, \dots, R_k\}$ is a finite nonempty set that includes the relations between individuals. If R is a b -ary relation, then $R \subseteq (I^S)^b$.
6. The size of schema \mathcal{S} is the size of individuals and is denoted by $|I^S|$.

3 Schema Matching

Schema matching is inherently heuristic. The schema matching approaches focus on obtaining the actual mappings between \mathcal{S} and \mathcal{T} . A mapping establishes a semantic correspondence between two individuals. For this reason, we introduce a heuristic definition of individual matching: if one or more labels of individual s in \mathcal{S} are semantically equivalent (mapped) to corresponding labels of individual t in \mathcal{T} , or the related relations are semantically equivalent, then we call that s and t are matched.

Definition 2. (Individual matching) If \mathcal{S} is the source schema, \mathcal{T} is the target schema, $s \in I^S$, $t \in I^T$, s and t are matched, such that:

1. There exists a function symbol f of arity a , $f^S(s, s_1, \dots, s_{a-1}) = l_i^S \Rightarrow f^T(t, t_1, \dots, t_{a-1}) = l_j^T$, which means that l_i^S is semantically equivalent to l_j^T , written $l_i^S \rightarrow l_j^T$. Where, $f \in \sigma$, $f^S \in F^S$, $f^T \in F^T$, $s_1, \dots, s_{a-1} \in I^S$, $t_1, \dots, t_{a-1} \in I^T$, $l_i^S \in Lab^S$, $l_j^T \in Lab^T$, or
2. There exists a relation symbol R of arity b , $R^S(s, s_1, \dots, s_{b-1})$ holds $\Rightarrow R^T(t, t_1, \dots, t_{b-1})$ holds, which means that the relation between s and s_1, \dots, s_{b-1} is equivalent to the relation between t and t_1, \dots, t_{b-1} , where, $R \in \sigma$, $R^S(s, s_1, \dots, s_{b-1}) \in R^S$, $R^T(t, t_1, \dots, t_{b-1}) \in R^T$, $s_1, \dots, s_{b-1} \in I^S$, $t_1, \dots, t_{b-1} \in I^T$.

Then, we write $s \rightarrow t$, or $\langle s, t \rangle$, which means there exists a mapping from s to t .

If the mapping of s and t only satisfy Condition 1, then it is called *label matching*; If the mapping only satisfy Condition 2, then it is termed *relation matching*; If the mapping satisfy two conditions at the same time, then it is called *structure matching*. In addition, for the mapping between s and t , the more semantic functions and relations can be matched, the *stronger* matching between s and t is.

Specifically, if $\forall f \in \sigma$, $f^S(s, s_1, \dots, s_{a-1}) = l_i^S \Rightarrow f^T(t, t_1, \dots, t_{a-1}) = l_j^T$, and if $\forall R \in \sigma$, $R^S(s, s_1, \dots, s_{b-1})$ holds $\Rightarrow R^T(t, t_1, \dots, t_{b-1})$ holds, then the mapping of s and t is the *strongest* matching.

In Equation 1, we show the individual matching result, where, $s \in I^S$, $t \in I^T$, ε stands for a void individual.

$$\text{matching result} \begin{cases} s \rightarrow t & \text{if } s \text{ and } t \text{ satisfy the conditions in Definition 2} \\ s \rightarrow \varepsilon & \text{if } s \text{ and } \forall t \in I^T \text{ dissatisfy the conditions in Definition 2} \end{cases} \quad (1)$$

Definition 3. (Schema matching) If \mathcal{S} is the source schema, \mathcal{T} is the target schema, the result of schema matching consists of all the mappings between I^S and I^T , written $\mathcal{S} \rightarrow \mathcal{T}$.

4 Schema Homomorphism

By the basic notion of *homomorphism* [5, 6], Definition 1, and Definition 2, we introduce the definition of *schema homomorphism*, which is the generalized definition of homomorphism.

Definition 4. A *schema homomorphism* $\varphi: \mathcal{S} \rightarrow \mathcal{T}$ from the source schema \mathcal{S} to the target schema \mathcal{T} is a mapping $\varphi: I^{\mathcal{S}} \rightarrow I^{\mathcal{T}}$ such that:

1. There exists a semantic function symbol f of arity n
 $f^{\mathcal{S}}(s_1, \dots, s_n) = l_n^{\mathcal{S}} \Rightarrow f^{\mathcal{T}}(\varphi(s_1), \dots, \varphi(s_n)) = \varphi(l_n^{\mathcal{T}})$
 for $s_1, \dots, s_n \in I^{\mathcal{S}}, l_n^{\mathcal{S}} \in \text{Lab}^{\mathcal{S}}$
2. There exists a semantic relation symbol R of arity m
 $R^{\mathcal{S}}(s_1, \dots, s_m) \text{ holds} \Rightarrow R^{\mathcal{T}}(\varphi(s_1), \dots, \varphi(s_m)) \text{ holds}$
 for $s_1, \dots, s_m \in I^{\mathcal{S}}$

If there exists a semantic homomorphism from \mathcal{S} to \mathcal{T} then we write $\mathcal{S} \rightarrow \mathcal{T}$.

Now, we prove that SMP is equivalent to finding whether there exists a semantic homomorphism from \mathcal{S} to \mathcal{T} , which preserves the semantics between two schemas.

Lemma 1. If \mathcal{S} and \mathcal{T} are matched, then there exists a semantic homomorphism from \mathcal{S} to \mathcal{T} .

Proof. By Definition 3, the result of schema matching consists of all the individual mappings. Without loss of generality, suppose that s is matched to t , $s \in I^{\mathcal{S}}, t \in I^{\mathcal{T}}$:

i) $\exists f \in \sigma$, $f^{\mathcal{S}}(s, s_1, \dots, s_{a-1}) = l_i^{\mathcal{S}} \Rightarrow f^{\mathcal{T}}(t, t_1, \dots, t_{a-1}) = l_j^{\mathcal{T}}$, f is a semantic function symbol of arity a , or ii) $\exists R \in \sigma$, $R^{\mathcal{S}}(s, s_1, \dots, s_{b-1}) \text{ holds} \Rightarrow R^{\mathcal{T}}(t, t_1, \dots, t_{b-1}) \text{ holds}$, R is a semantic relation symbol of arity b . For every mapping, let $\varphi(s) = t$, $\varphi(l_i^{\mathcal{S}}) = l_j^{\mathcal{T}}$, by Definition 4, Lemma 1 is completed.

Lemma 2. If there exists a semantic homomorphism from \mathcal{S} to \mathcal{T} , then \mathcal{S} and \mathcal{T} are matched.

Proof. If there is a semantic homomorphism from \mathcal{S} to \mathcal{T} , by Definition 4, then:

- i) $\exists f \in \sigma$, $f^{\mathcal{S}}(s, s_1, \dots, s_{n-1}) = l_n^{\mathcal{S}} \Rightarrow f^{\mathcal{T}}(\varphi(s), \varphi(s_1), \dots, \varphi(s_{n-1})) = \varphi(l_n^{\mathcal{T}})$;
- ii) $\exists R \in \sigma$, $R^{\mathcal{S}}(s, s_1, \dots, s_{m-1}) \text{ holds} \Rightarrow R^{\mathcal{T}}(\varphi(s), \varphi(s_1), \dots, \varphi(s_{m-1})) \text{ holds}$.

Let $\varphi(s) = t$, $\varphi(s_i) = t_i$, $\varphi(l_n^{\mathcal{S}}) = l_n^{\mathcal{T}}$, by Definition 2, we obtain s is matched to t , and s_i is matched to t_i , all mappings constitute the result of schema matching.

Based on Lemma 1 and 2, we show the theorem of SMP in homomorphism.

Theorem 1. Two schemas \mathcal{S} and \mathcal{T} are matched iff there exists a schema homomorphism from \mathcal{S} to \mathcal{T} , $\mathcal{S} \rightarrow \mathcal{T}$.

Proof. \Leftarrow Lemma 2, if $\mathcal{S} \rightarrow \mathcal{T}$, then \mathcal{S} and \mathcal{T} are matched.

\Rightarrow Lemma 1, if \mathcal{S} and \mathcal{T} are matched, then $\mathcal{S} \rightarrow \mathcal{T}$.

Now, we formulize SMP as the SHOM problem, and build the algebraic framework for schema matching, and obtain the algorithm model of SMP.

Algorithm Model of SMP: Given two schemas \mathcal{S} and \mathcal{T} , the goal of matching algorithms is to find the semantic homomorphism between \mathcal{S} and \mathcal{T} .

5 Taxonomy of Schema Matching

In Definition 1, the label collection includes all kinds of labels of individuals, such as name label, concept label, and attribute label, etc. The label classification is convenient for practitioner to design and analyze the different matching approaches of SMP [13]. In Fig. 1, we present the matching methods based on the types of label symbols.

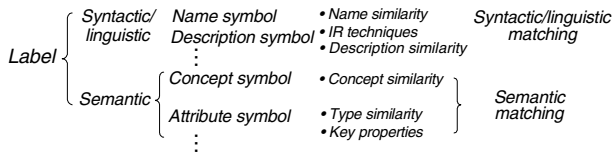


Fig. 1. Matching methods based on the different types of labels

We prove that SMP is equivalent to finding the semantic homomorphism from the source schema to the target schema, therefore, we can call schema matching as *homomorphic matching*. Here, we show the overall classification of SMP by SHOM framework, which is based on the classification in [13].

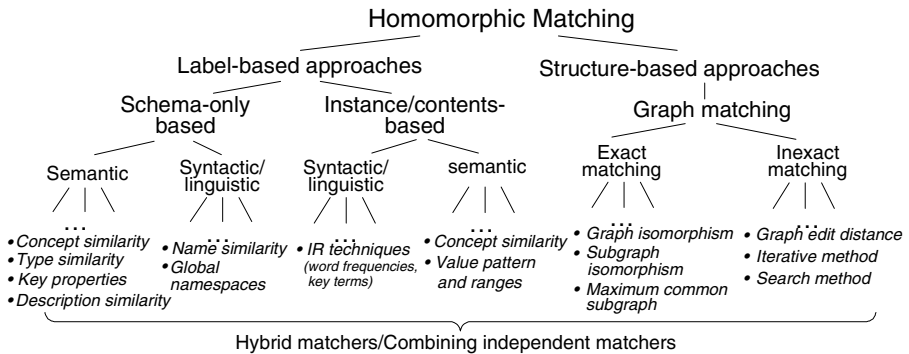


Fig. 2. Classification of schema matching approaches

6 Conclusions and Future Work

It is well known that a formal framework for SMP is important because it facilitates the building of algorithm model and the evaluation of algorithms. This paper is intended to develop an algebraic framework for generic schema matching. We have

three contributions: Firstly, since the schemas are the finite structures over the specific signatures, we propose a meta-meta model of schema, which called *generic matching-oriented* model. This definition has a distinctive feature: it is able to describe any particular style of schemas, and transforms a schema and other matching information into a finite structure over specific signature. The signature is a collection of individual, label, relation and function symbols. Secondly, we demonstrate that SMP is equivalent to finding the schema homomorphism between two schemas. Then, the algebraic framework for generic schema matching is developed, which is the main contribution of this paper. Thirdly, based on Definition 2 (individual matching) and Definition 4 (schema homomorphism), we show a new classification of schema matching. This algebraic framework is able to characterize SMP elegantly.

Homomorphism is a useful model for a wide variety of combinatorial problems dealing with mappings and assignments [6]. In this framework, SMP is transformed into a semantic homomorphism problem, which can guide practitioner to design the effective algorithms for SMP and evaluate the algorithms. We can use the approaches of combinatorial optimization, such as neural network, machine learning, and local search, etc., to solve $n:m$ matching problem [6, 9, 11]. These optimization approaches are widely used to solve graph homomorphisms or graph matching problems [2].

References

1. Alagić, S., Bernstein, P. A.: A model theory for generic schema management, Springer-Verlag, LNCS 2497: 228-246.
2. Bunke, H.: Graph matching: Theoretical foundations, algorithms, and applications, in Proc. Vision Interface 2000, 82 - 88
3. Doan, A., Domingos, P., Halevy, A.: Learning to Match the Schemas of Data Sources: A Multistrategy Approach. Machine Learning. Kluwer Academic Publishers Manufactured in The Netherlands, 2003 (50): 279-301.
4. Dubhashi, D. P.: Complexity of Logical Theories. Department of Computer Science, University of Aarhus. BRICS LS-95-5, ISSN 1395-2048, 1995
5. Federa, T., Madelaineb, F., Stewartc, I. A.: Dichotomies for classes of homomorphism problems involving unary functions, Theoretical Computer Science 2004 (314): 1- 43.
6. Hell, P.: Algorithmic aspects of graph homomorphisms, Combinatorics 2003, London Math. Society Lecture Note Series 307, Cambridge University Press, 239-276.
7. Hull, R.: Managing semantic heterogeneity in databases: A theoretical perspective, PODS 1997, 51-61.
8. Madhavan, J., Bernstein, P. A., Rahm, E.: Generic schema matching with cupid, VLDB 2001.
9. Melnik, S.: Generic model management-concepts and algorithms, Springer, LNCS 2967, 2004.
10. Miller, R. J., Ioannidis, Y. E., Ramakrishnan, R.: Schema equivalence in heterogeneous systems: Bridging theory and practice, Information Systems 19(1): 3-31, 1994.
11. Mugnier, M. L.: Knowledge representation and reasonings based on graph homomorphism, Springer, LNAI 1867: 172-192.
12. Paolini, P., Pelagatti, G.: Formal Definition of Mappings in a Data Base. In Proc. ACM SIGMOD Intl. Conf. on Management of Data, 1977:40- 46.
13. Rahm, E., Bernstein, P. A.: A survey of approaches to automatic schema matching, The VLDB Journal, 2001(10): 334-350.

A Clustering Algorithm Based Absorbing Nearest Neighbors¹

Jian-jun Hu¹, Chang-jie Tang¹, Jing Peng^{1,2}, Chuan Li¹, Chang-an Yuan^{1,3},
and An-long Chen¹

¹School of Computer Science, Sichuan University, Chengdu 610064, China
hujianjun@cs.scu.edu.cn, tangchangjie@vip.sina.com

²Department of Science and Technology, Chengdu Public Security Bureau,
Chengdu 610017, China

³Department of Information & Technology, Guangxi Teachers Education University,
Nanning, Guangxi 530001, China

Abstract. The clustering over various granularities for high dimensional data in arbitrary shape is a challenge in data mining. In this paper Nearest Neighbors Absorbed First (NNAF) clustering algorithm is proposed to solve the problem based on the idea that the objects in the same cluster must be near. The main contribution includes : (1) A theorem of searching nearest neighbors (SNN) is proved. Based on it, SNN algorithms are proposed with time complexity $O(n \cdot \log(n))$ or $O(n)$. They are much faster than the traditional searching nearest neighbors algorithm with $O(n^2)$. (2)The clustering algorithm of NNAF to process high dimensional data with arbitrary shape is proposed with time complexity $O(n)$. The experiments show that the new algorithms can process efficiently high dimensional data in arbitrary shape with noisy. They can produce clustering over various granularities quickly with little domain knowledge.

1 Introduction and Background

Clustering Analysis is an important way in knowledge discovery fields. A cluster is a collection of data objects with higher similarity within cluster and lower similarity between clusters. The degree of similarity is usually described by the distance between objects. The greater is the distance, the smaller the similarity is, vice versa. An ideal clustering algorithm should possess scalability, discovery of clusters with arbitrary shape, minimal requirement for input parameters, insensitive to noisy data, ability to deal with high dimension data, interpretability and usability. A lot of clustering algorithms have been proposed in the past years. The existing methods can be classified into the following categories: partitioning methods, hierarchical methods, density-based methods, grid-based methods, model-based methods [1]. The typical algorithms are K-means algorithm [2], CURE algorithm [3], DBSCAN algorithm [4], CLIQUE algorithm [5], BIRCH algorithm [6] [7], etc.. They solve some specific problems by specific methods. However, the multi-layer clustering of high dimensional data with arbitrary shape is also a challenge research fields.

¹ Supported by Grant of National Science Foundation of China (60473071), and Specialized Research Fund for Doctoral Program by the Ministry of Education (SRFDP 20020610007).

This paper proposes an improved nearest distance clustering algorithm—Nearest Neighbors Absorbed First (NNAF) based on the idea of objects in the same cluster must be near.

2 Related Works

The hierarchical clustering algorithms can be classified as being either bottom-up approach or top-down approach. The former starts with each object forming a separate group. Then it successively merges the objects or groups close to one another, until all of the groups are merged into one (the topmost level of the hierarchy), or until a termination condition holds. The latter begins with all the objects in the same cluster. In each successive interaction, a cluster is split into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds [1]. The typical hierarchical clustering algorithms are BIRCH algorithm, CURE algorithm, Shortest Distance algorithm [8], CHAMALEON algorithm [9], and so on so forth.

There are some common features among NNAF algorithm and Shortest Distance algorithm. In the Shortest Distance algorithm, in order to get the nearest neighbors of an object, you have to compare the distance from it to all other else. Its time complexity is $O(n^2)$. Thus it is not fit for the clustering with large numbers of objects. And it can not process noisy data efficiently. With these statements in mind, NNAF algorithm that can process efficiently clustering with arbitrary shape and noisy data is proposed in this paper. It succeeds in the virtues of Shortest Distance algorithm, which can process high dimensional objects efficiently and nearly require little domain knowledge.

3 Nearest Neighbors Absorbed Firstly (NNAF) Algorithm

3.1 Definitions

Definition 1. Given a dataset V in a high dimensional space and distance threshold d , where $d > 0$, $V = \{p_1, p_2, \dots, p_n\}$, then

- (1) The distance between p_1 and p_2 is donated as $D(p_1, p_2)$;
- (2) If $D(p_1, p_2) < D(p_1, p_3) < \dots < D(p_1, p_n)$, then p_2 is the nearest object of p_1 , namely p_2 is the nearest neighbor of p_1 , which is donated as $NN(p_1) = p_2$;
- (3) If $NN(p_1) = p_2$ and $D(p_1, p_2) \leq d$, then p_2 and p_1 are of the same cluster.

The basic idea of NNAF algorithm is classifying the two objects satisfying $NN(p_1) = p_2$ or $NN(p_2) = p_1$ into the same cluster. Let $NN(p_1) = p_2$ and $D(p_1, p_2) \leq d$, where d is the distance threshold. If p_1 is of the first cluster and p_2 belongs to no cluster, then p_2 also belong to the first cluster, vice versa. If p_1 is of the first cluster and p_2 is of the second cluster, then the two clusters recombine into one new cluster and all objects that belong to the two clusters are classified into the new cluster.

Definition 2. Given a dataset $V = \{p_1, p_2, \dots, p_n\}$, where $p_i \in V, p_k \in V$:

- (1) Each object of V has a 5-tuple $(Dir, Distance, Cluster, Neighbors, Reverse-Neighbors)$, in the terms of p_i , where Dir is the coordinates of the point p_i ; $Distance$ is the distance between p_i and its nearest neighbor; $Cluster$ is the cluster of p_i ; $Neighbors$ is the set of nearest neighbors of p_i ; $Reverse-Neighbors$ is the set of objects whose nearest neighbor is p_i .
- (2) If $NN(p_i) = p_k$ and $NN(p_k) = p_i$, then p_i and p_k are said to be the nearest neighbor to each other.
- (3) If $NN(p_i) = p_k$ and the attributes of cluster of p_k is empty, then p_k is said to be the unclassified point of p_i . Otherwise p_k is the classified point of p_i .

Emerging nearest neighbor to each other or classified nearest neighbor is the clustering termination condition of NNAF.

3.2 Nearest Neighbors Searching (SNN) Algorithm

To avoid comparing the distance between one object to all other else when searching its nearest neighbors, our new SNN algorithm tends to find the nearest neighbors fast. Using SNN algorithm, the nearest neighbors can be found by only comparing the distances from one object to the objects in its δ domain.

Theorem 1. Let $V = \{p_1, p_2, \dots, p_i, \dots, p_m\}$ be a dataset in n -dimensional space, where $p \in V, p_i$ is a object in V . Let d be the distance threshold. Each dimension of the object with the distance to p less than d must be within d domain in the corresponding dimension of p .

Corollary 1. Suppose p and q are two objects in n -dimensional space and the project of the distance from p to q on a dimension is d . If p is in the d domain region of q , namely $D(p, q) \leq d$, then the coordinates of p is same as that of q on other dimensions.

The space of searching can be reduced dramatically in searching nearest neighbor algorithm based on corollary 1.

Algorithm 1. Searching Nearest Neighbors (SNN)

Input: Dataset $V = \{p_1, p_2, \dots, p_i, \dots, p_m\}$, where all objects has been sorted by a certain-dimension, which is denoted by sorted dimension; distance threshold d .

Output: The 5-tuple attributes of all objects, where $Cluster = NULL$, the objects in the attributes of $Neighbors$ and $Reverse-Neighbors$ is that the distance holds the distance threshold.

By SNN, it does not need calculate the distance from one object to all others and only compare the distance from it to the objects in a small domain of it. Thus the computing speed is increased greatly. The time complexity of SNN is $O(n * \log(n))$. If the objects are obtained by scanning image, the data has been sorted after scanning. So the time complexity will be $O(n)$. However, that of Shortest Distance algorithm by comparing all distances between every two objects is $O(n^2)$. After user adjusted the distance threshold d , the algorithm only needs search the nearest neighbors of outliers that have been found in the last time by SNN.

3.3 Nearest Neighbors Absorbed Firstly (NNAF) Algorithm

The idea of NNAF is described as follows: At first, an object Obj1 is classified into a cluster, then all nearest neighbors of Obj1 and the objects whose nearest neighbor is Obj1 are added in the cluster. Finally, all the nearest neighbors of the objects that have been added in the cluster and the objects whose nearest neighbors are the objects that have been added in the cluster are added in the cluster. NNAF iteratively does this process until no objects add in this cluster. In other words, all the nearest neighbors of the objects in this cluster and the objects whose nearest neighbors are the object in this cluster have been added in this cluster. At this time, another unclassified object is classified into a new cluster and performing the clustering follows the above way, until all objects have been added in a certain cluster. The clustering is finished.

Algorithm 3. NNAF algorithm

Input: dataset $V = \{p_1, p_2, \dots, p_n\}$, the distance threshold d , the quantity threshold q ;

Output: the attribute Cluster of every object

In this algorithm, SNN algorithm is invoked first. Then NNAF algorithm executes following the idea described above.

In NNAF algorithm, getting the clusters need scan the database only once with the time complexity $O(n)$. And the time complexity of finding the nearest neighbors is $O(n \cdot \log(n))$. Therefore the total time complexity is $O(n \cdot \log(n))$. When the objects are obtained by scanning image, the time complexity will decrease to $O(n)$.

3.4 Algorithm Analysis

The algorithm proposed in this paper need little domain knowledge because it only needs two thresholds. The values of the thresholds can be tried many times until the satisfied clustering is obtained.

The clustering based on the nearest neighbors in NNAF has nothing to do with the distributed shape of the objects, the first object selected and the dimensionality. And it needs little domain knowledge and decreases the difficulty for choosing parameters. Therefore, the algorithm fits for clustering with arbitrary shape and high dimensionality. The time complexity of the algorithm is $O(n)$ since it scans the database only once. The sum of the time complexity is $O(n \cdot \log(n))$ after adding the searching nearest neighbors time.

4 Experiments and Results

All of our experiments are conducted on a PC with Intel Pentium III 1GHZ processor and 256 MB memory, which runs Windows XP professional operation system. The initial data of 2-dimensional is shown as Figure 1. It is obviously there are many noisy data and the data distributed with arbitrary shape. It is difficult for many traditional clustering algorithms to process this kind of data. The clustering result is shown as Figure 2 by NNAF.



Fig. 1. Dataset



Fig. 2. The clustering result by NNAF

In this experiment the distance threshold is 2. When it is over 2 the clustering result is nearly the same one. It is because the nearest neighbors can not change however the distance threshold increases when the distance from data to their nearest neighbors are less then the distance threshold. When the distance from most data to their nearest neighbors are more then the distance threshold, the clustering result will be changed dramatically by increasing the distance threshold.

The experimental results show that the algorithms proposed in this paper can solve efficiently the clustering problem with arbitrary shape and noisy data.

Two methods of searching nearest neighbors are used in order to evaluate the performance of SNN. The first one is SNN and the second one is the traditional method. The traditional method is the method which finds the nearest neighbors of every object by comparing all the distances from it to all the other else. The experimental results are shown in Table 1. In Table1 the first column is the number of data, the second and the third columns are the time being used by the two methods respectively. The time unit is millisecond. We can see that SNN is better than the traditional algorithm. It is should be stated that the data comes from scanning Figure 1. The sorting time is saved since the data have been sorted by coordinates after scanning. Although the sorting time is add the SNN is also faster than the traditional algorithm. Because the time complexity of traditional algorithm is $O(n^2)$ and that of the SNN is $O(n \cdot \log(n))$.

Table 1. Time of searching nearest neighbors

Data Quantity	Time of SNN (ms)	Time of Traditional algorithm (ms)
2000	39	5587
4000	69	10825
6000	100	17314
8000	129	23052
10000	160	28892
10624	179	29161

5 Conclusion

In summary, a serial of algorithms about clustering on higher dimension and some related theorems are proposed in this paper. The experiment results indicate that the algorithms are reasonable and can produce satisfied clustering over various granularities. NNAF is also an effective clustering algorithm.

References

- [1] Han JW, Kambr M. *Data Mining Concepts and Techniques*. Beijing: Higher Education Press, 2001. 145~176.
- [2] Kaufan L, Rousseeuw PJ. *Finding Groups in Data: an Introduction to Cluster Analysis*. New York: John Wiley & Sons, 1990.
- [3] Guha S, Rastogi R, Shim K. CURE: an efficient clustering algorithm for large databases. In: Haas LM, Tiwary A, eds. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. Seattle: ACM Press, 1998. 73~84.
- [4] Ester M, Kriegel HP, Sander J, Xu X. A density based algorithm for discovering clusters in large spatial databases with noise. In: Simoudis E, Han JW, Fayyad UM, eds. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. Portland: AAAI Press, 1996. 226~231.
- [5] Agrawal R, Gehrke J, Gunopulos D, Raghavan P. Automatic subspace clustering of high dimensional data for data mining application. In: Haas LM, Tiwary A, eds. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. Seattle: ACM Press, 1998. 94~105.
- [6] Tian Zhang, Raghu Ramakrishnan and Miron Livny. BIRCH: An Efficient Data Clustering Method for Very Large Database, Technical Report, Computer Sciences Dept., Univ. of Wisconsin-Madison, 1995
- [7] Zhang, T., Ramakrishnan, R., Livny, M. BIRCH: an efficient data clustering method for very large databases. In: Jagadish, H.V., Mumick, I.S., eds. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*. Quebec: ACM Press, 1996. 103~114.
- [8] Beyer, K.S., Goldstein, J., Ramakrishnan, R., et al. When is 'nearest neighbor' meaningful? In: Beeri, C., Buneman, P., eds. *Proceedings of the 7th International Conference on Data Theory, ICDT'99*. LNCS1540, Jerusalem, Israel: Springer, 1999. 217~235.
- [9] Karypis, G., Han, E.H., Kumar, V. CHAMELEON: a hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, 1999, 32 (8):68~75.

Parallel Mining of Top-K Frequent Itemsets in Very Large Text Database*

Yongheng Wang, Yan Jia, and Shuqiang Yang

Institute of Network, Computer School, National University of Defense Technology,
Changsha, China

TommyWang3465@hotmail.com
jiayanjy@vip.sina.com
sqyang9999@163.com

Abstract. Frequent itemsets mining is a common and useful task in data mining. But most of the current mining algorithms can't be used in very large text database. In this paper, we propose a novel and efficient parallel algorithm parTFI which is used to find top-k frequent itemsets with specified minimum length in very large text database. Base on a simple data structure H-struct, parTFI uses a novel logical vertical data partition technique to mine top-k frequent itemsets at each mining server parallel. Our performance study shows that when processing very large sparse text database, parTFI outperforms Apriori and FP-growth, two efficient frequent itemsets mining algorithms, even when both are running with the better tuned *min_support*. Furthermore, by creating H-struct dynamically, parTFI can suit even huge dataset that most other algorithms can't process.

1 Introduction

As an important data mining problem, frequent itemsets mining plays an essential role in many data mining tasks, such as mining association rules, classification, and clustering. Recently, frequent itemset is widely used in text mining, such as text classification [1] and text clustering [2].

There have been many algorithms developed for efficient mining of frequent itemsets, which can be classified into 3 categories: (1) Apriori-based, horizontal formatting method, with Apriori [3] as its representative, (2) Apriori-based, vertical formatting method, such as CHARM [7], and (3) projection-based pattern growth method, which may explore some compressed data structure such as FP-tree, as in FP-growth [4]. The common idea in such algorithms is to use a *min_support* threshold to ensure the generation of the correct and complete set of frequent itemsets. But when used in very large text database, frequent itemsets (also named frequent termsets in this domain) mining encounters new challenges. (1) Text data is usually high dimensional and sparse which makes most common frequent itemsets mining algorithms inefficient. (2) Setting *min_support* is a difficult task. (3) Frequent itemset mining often

* This project is sponsored by national 863 high technology development foundation (No.2004AA112020, No.2003AA115210 and No.2003AA111020).

leads to the generation of large number of patterns, including short patterns and long patterns. However, unlike in other area, short frequent itemsets is often of no use in text area. (4) Text database can be very large.

The second problem can be solved by mining top-k frequent itemsets without setting *min_support*. The last problem can be solved by using disk-based algorithms instead of memory-based algorithms or using parallel and distributed mining algorithms [6].

Based on the observation above, we provide the following task of finding frequent itemsets in very large text database: mining top-k frequent itemsets with minimum length min_l , where k is the desired number of frequent patterns to be mined, and min_l is the minimal length of each itemset.

2 Problem Definition and Related Work

A transaction database TDB is a set of transactions, where each transaction, denoted as a tuple $\langle tid, X \rangle$, contains a set of items (i.e., X) and is associated with a unique transaction identity tid. Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of items. An itemset Y is a non-empty subset of I. The length of itemset Y is the number of items contained in Y, and Y is called an l -itemset if its length is l . An itemset Y is contained in transaction $\langle tid, X \rangle$ if $Y \subseteq X$. Given a transaction database TDB, the support of an itemset Y, denoted as $sup(Y)$, is the number of transactions in TDB which contain Y.

Definition 1. (top-k frequent itemsets) An itemset Y is a frequent itemset if $sup(Y) \geq min_support$. A frequent itemset Y is a top-k frequent itemset of minimal length min_l if there exist no more than $(k - 1)$ itemsets of length at least min_l whose support is higher than that of Y. \square

Many algorithms such as Apriori and FP-growth [8] can be applied to this problem. Apriori has to join all the length l itemsets to generate length $l + 1$ candidate for all l from 1 to min_l which makes it inefficient. FP-growth shows better performance and is widely used nowadays. But the FP-growth algorithm is not easy to be parallelized and can't suit very large dataset since it needs too much memory when the dataset is very large. The author of FP-growth also pointed out this algorithm does not perform well on sparse data [5] and proposed a new pattern-growth algorithm named H-mine [9]. In this paper, we propose a novel parallel algorithm, named parTFI (Parallel Top-k Frequent Itemset), which is based on H-struct in [9] and can efficiently find top-k frequent itemsets of minimal length min_l in very large sparse database. A controlling server and n mining servers are used in parTFI.

3 Method Development and parTFI Algorithm

3.1 Method Development

The parTFI algorithm is based on H-struct which is described in [9]. Since we only need itemsets with minimum length min_l , we can reduce the size of the H-struct and improve the performance of the mining algorithm based on the following remarks:

Remark 1. (Absolute short transaction) If a transaction T contains less than min_l distinct items, T is called an absolute short transaction, since none of the items in T can contribute to an itemset of minimum length min_l . □

Remark 2. (Relative short transaction) Let I be a frequent itemset has been found currently, i be the item linked by the header table H_i in transaction t. The length of I is l_i and the length from i to the end of transaction t is l_left . If $l_i + l_left < min_l$, t is called a relative short transaction and it can't contribute to the frequent itemset begin with I. □

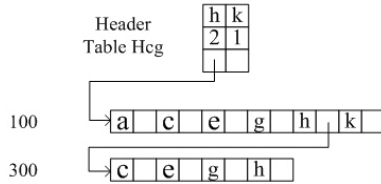


Fig. 1. relative short transaction

For example, in Fig. 1, transaction 300 is a relative short transaction. The absolute short transaction remark can be used when the H-struct is constructed and the relative short transaction remark can be used when mining the H-struct for frequent itemsets.

The basic idea of our algorithm at each mining server is to set $min_support = 0$ at the beginning and construct a full H-struct. When mining frequent itemsets in the H-struct, a simple data structure called K-itemset table as shown in table 1 is used. The current top-k frequent itemsets is stored in the table sorted by the support of the itemset.

Table 1. A k-itemset table

Num	Itemset	Support
1	a c h k	100
2	a c e g h k	70
..
k	b g k m	55

Remark 3. (Header table pruning) The minimum support in the k-itemset table is safe to be used to prune the header table. □

To support large dataset, data partitioning technology is used. Usually data is partitioned into n partitions and each partition is processed in a mining server in parallel mining algorithms. We call this horizontal partition.

In parTFI, we use a novel logical vertical partition according to subset of the resulting frequent itemsets. As described in [9], the result of H-mine is partitioned into subsets and there is no overlap between subsets. So we can partition the data logically according to the resulting subsets. For example, suppose we have six items from a to

f, we call the resulting subset containing item a a-subset, that containing b but not a b-subset, and so on. We can partition the six subsets into 2 partitions as shown in Fig.3.

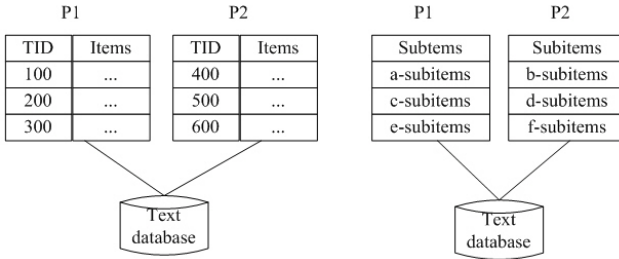


Fig. 2. traditional horizontal partition **Fig. 3.** logical vertical partition

Remark 4. (Global top-k frequent itemsets) Using the logical vertical partition, the global top-k frequent itemsets must be in the local top-k frequent itemsets. □

Since there is no overlap in all subsets in the logical vertical partition, the local support of each frequent itemset is the same as the global support of that itemset. So the global top-k frequent itemsets must be in the local top-k frequent itemsets.

3.2 parTFI Algorithm

Now we summarize the entire mining process and present the parTFI algorithm.

Algorithm 1. Parallel mining of top-k itemsets with minimal length min_l in a very large text database.

Algorithm for Controlling Server:

Input: (1) A very large text database, (2) an integer k, i.e., the k most frequent itemsets to be mined, and (3) min_l , the minimal length of the frequent itemsets.

Output: The set of the frequent itemsets which satisfy the requirement.

Method:

1. Logical vertical data partition. Partition the resulting subsets into n partitions named P1, P2, ... Pn;
2. Notify each mining server to run with input (P_i, k, min_l);
3. Wait for the result of all mining server. When all result is returned, sort the result and return the top-k frequent itemsets.

Algorithm for Mining Server:

Input: (1) A logical vertical partition, (2) an integer k, i.e., the k most frequent itemsets to be mined, and (3) min_l , the minimal length of the frequent itemsets.

Return: The local top-k frequent itemsets in the partition.

Method:

1. Initially, $min_support$ is set to 0;
2. Create an empty k-itemset table;
3. Construct the H-struct. Absolute short transactions are removed during the constructing process;

4. Begin mining the H-struct to find frequent itemset. During this process:
 - a) Relative short transactions remark is used to improve the mining performance;
 - b) Each frequent itemset has been found is inserted into the k-itemset table;
 - c) If the minimum support in the k-itemset table is large than $min_support$, then:
 - i. set $min_support$ = the minimum support in the k-itemset table;
 - ii. Prune the header table in the H-struct according to $min_support$;
5. When the whole H-struct is processed, send the k-itemset table to the controlling server. □

4 Experimental Evaluation

In this section, we report our performance study of parTFI. We compare the efficiency of parTFI with two other frequent itemsets mining algorithms: FP-growth, currently considered one of the fastest algorithm for frequent itemsets mining, and the Christian Borgelt’s implementation of Apriori¹. We provide traditional horizontal partition for FP-growth and Apriori since they can’t run in the logical partition mode like parTFI. We run these two algorithms in each mining server to find frequent itemsets in a partition of the database. To give the best credit to FP-growth and Apriori, we also assign best tuned $min_support$ to these two algorithms to make sure they can generate frequent itemsets with reasonable size and then sort the itemsets to find the top-k frequent itemsets.

The datasets used in our experiments can be grouped into the following two categories: (1) Sparse real dataset which is collected from public forums in the internet by robots. (2) Dense synthetic dataset which is generated from a set of seeds.

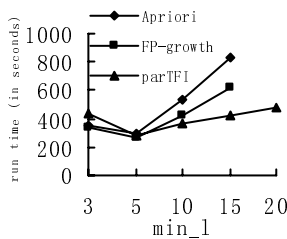


Fig. 4. sparse dataset, k = 200

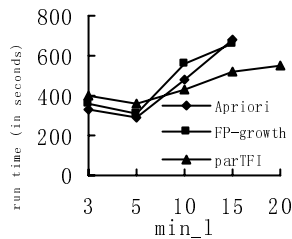


Fig. 5. sparse dataset, k = 500

We use 4 HP demonstration units connected with fast Ethernet. Each unit has 4 Itanium 2 1.3G processors and 10 GB memory. The text data is stored in an oracle 10g database (4 node cluster).

¹ <http://fuzzy.cs.uni-magdeburg.de/~borgelt>

We first compare the performance of parTFI with FP-growth and Apriori on the real sparse dataset with size 4GB. parTFI performs consistently better than FP-growth and Apriori if the min_l is not too small. Fig. 4 and fig. 5 shows the running time of these three algorithms on real sparse dataset. In the next experiment, we compare these three algorithms on dense synthetic dataset. As shown in fig. 6 and 7, parTFI has comparable performance with FP-growth and is better than Apriori. We have also studied the performance of parTFI on 40GB sparse text dataset. As shown in Fig. 9, parTFI can well suit very large text dataset which most other algorithm can't process.

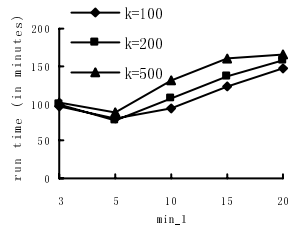
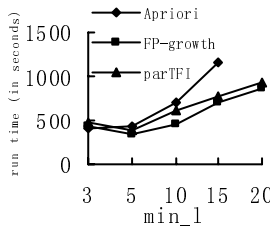
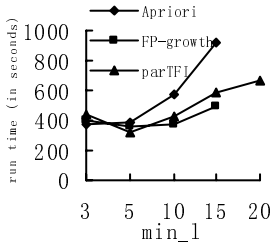


Fig. 6. dense dataset, k = 200 Fig. 7. dense dataset, k = 500 Fig. 8. massive sparse dataset

From these experiments we can conclude that parTFI is efficient for mining top-k frequent itemsets with a specified minimum length in very large sparse dataset. Unlike Apriori and FP-growth whose performance deteriorates as min_l increases, parTFI's running time almost stays low. In addition, parTFI has better scalability since it is a parallel and distributed algorithm and it uses dynamic H-struct construction technique for very large database.

5 Discussion and Conclusion

The parTFI represents a novel, highly efficient and scalable mining method for large text database. Parallel and distributed mining is a reasonable way to find top-k frequent itemsets in very large database. More detailed study along this direction is needed, including further improvement of the performance by applying other heuristics to pruning the header table and using new method to reducing the number of local frequent itemsets need to be mined.

References

1. Maria-Luiza Antonie, Osmar R. Zaiane.: Text Document Categorization by Term Association. In: Proc. of the IEEE 2002 International Conference on Data Mining (ICDM'2002), pp 19-26, Maebashi City, Japan, December 9 - 12, 2002
2. Beil, F., Ester, M., Xu, X.: Frequent Term-Based Text Clustering. ACM SIGKDD, 2002.
3. R. Agrawal and R. Srikant.: Fast algorithms for mining association rules. VLDB'94.
4. J. Han, J. Pei, and Y. Yin.: Mining frequent patterns without candidate generation. SIGMOD'00.

5. Z. Zheng, R. Kohavi, and L. Mason.: Real World Performance of Association Rule Algorithms. In: Proc. of KDD-2001, 2001.
6. R. Agrawal and J. Shafer.: Parallel and Distributed Association Mining: A Survey. "Parallel Mining of Association Rules," IEEE Trans. Knowledge and Data Eng., Vol. 8, No. 6, Dec. 1996, pp. 962 - 969.
7. Zaki MJ, Hsiao CJ.: CHARM: An efficient algorithm for closed itemset mining. In: Grossman R, et al, eds. Proc. of the 2nd SIAM Int'l. Conf. on Data Mining. Arlington: SIAM, 2002. 12~28.
8. J. Han, J. Pei, and Y. Yin.: Mining frequent patterns without candidate generation. In: SIGMOD'00, pages 1 - 12.
9. J. Pei, J. Han, H. Lu, S. Nishio, and D. Tang, S. amd Yang.: H-Mine: Hyper-Structure Mining of Frequent Patterns in Large Databases. In: Proc. of the 2001 IEEE ICDM Conf., San Jose, CA, USA, 2001.

Removing Smoothing from Naive Bayes Text Classifier

Wang-bin Zhu¹, Ya-ping Lin¹, Mu Lin², and Zhi-ping Chen¹

¹ Computer and Communication College, Hunan University,
Hunan Changsha, 410082, China

zhuwangbin@hotmail.com, {yplin, jt_zpchen}@hnu.cn

² Mathematics and Econometrics College, Hunan University,
Hunan Changsha, 410082, China

kevin9908@hotmail.com

Abstract. Smoothing is applied in Bayes classifier when the maximum likelihood (ML) estimate can't solve the problem in the absence of some features in training data. However, smoothing doesn't have firm theoretic base to rely on as ML estimate does. In this paper, we propose two novel strategies to remove smoothing from the classifier without sacrificing classification accuracy: NB_TF and NB_TS. NB_TF adjusts the classifier by adding the test document before classification and it is suitable for online categorization. NB_TS improves the performance by adding the whole test set to the classifier in the training stage and it is more efficient for batch categorization. The experiments and analysis show that NB_TS outperforms Laplace additive smoothing and Simple Good-Turing (SGT) smoothing, and NB_TF performs better than Laplace additive smoothing.

1 Introduction

Text categorization (TC), the activity of labeling natural language texts with thematic categories from a predefined set, has gained a prominent status in the information systems field, largely due to the widespread and ever growing availability of digital documents and the consequential need on the part of the users to access them in flexible ways [1,2].

Due to the simplicity and good performance, Naive Bayes (NB) classifier has been gaining popularity lately. There are two event models of NB: multivariate Bernoulli model and multinomial model. The latter is used more widely in TC [3]. Recently, the focus on NB is to improve the accuracy by alleviating the skewed data bias, performing heuristic feature transformations, normalizing by the length of the documents and taking the logarithms of the counts [4,5]. Smoothing is commonly used to avoid zero probability estimates, and plays an important role in NB. There are several smoothing methods for statistical language modeling, such as additive smoothing, the GT estimate, Katz smoothing etc [6]. Two most popular smoothing methods used in NB are additive smoothing and GT smoothing. Experiments show GT smoothing is more competent than

additive smoothing in web document categorization [7]. William A. Gale introduced the Linear Good-Turing (LGT) estimate, proposed Simple Good-Turing (SGT) estimate which was proven to be a good scheme[8].

In spite of improving categorization accuracy in some extent, smoothing suffers many negative effects. First, smoothing doesn't have firm theoretic base to rely on as ML estimate does [9]. In addition, some smoothing strategies used in Bayes classifier are neither efficient nor easy to understand. For example, GT smoothing is difficult to understand and not very efficient because of the complicated smoothing process [10]. Laplace additive smoothing does not consider out of vocabulary (OOV) features [11], and one can often do better by explicitly considering them. Hence, if we can avoid smoothing in NB without sacrificing categorization accuracy, the classifier will be more effective and more efficient.

In this paper, we propose two new strategies, NB_TF and NB_TS, to remove smoothing from NB without sacrificing classification accuracy. Compared with Laplace smoothing and SGT smoothing, NB_TS performs the best and NB_TF outperforms Laplace smoothing but inferior to SGT.

The rest of the paper is organized as follows. In Section 2 we review the NB algorithm and related smoothing methods. In Section 3, we discuss NB_TF and NB_TS. Section 4 gives the experimental results and analysis. Finally, Section 5 concludes the paper.

2 Related Works

2.1 Naive Bayes Algorithm

NB computes the posterior probabilities that the document $\vec{d}_i = (d_{i1}, \dots, d_{in})$ belongs to different classes and assigns it to the class with the highest probability.

$$\text{class}(d_i) = \arg \max_{1 \leq k \leq |C|} \{P(c_k|d_i)\} = \arg \max_{1 \leq k \leq |C|} \{P(c_k)P(d_i|c_k)\} . \quad (1)$$

$$P(c_k) = \frac{\sum_{i=1}^{|D|} P(c_k|d_i)}{|D|} . \quad (2)$$

$$P(d_i|c_k) = \prod_{j=1}^n P(t_j|c_k)^{d_{ij}} . \quad (3)$$

The computation of $P(t_j|c_k)$ in (3) varies according to the event model chosen for document representation and the smoothing method adopted.

2.2 Smoothing Methods

There are many smoothing methods for NB, among which additive smoothing and GT smoothing are used most widely.

Additive Smoothing. The $P(t_j|c_k)$ required in (3) is calculated as follows:

$$P(t_j|c_k) = \frac{\delta + \sum_{d_i \in c_k} d_{ij}}{\delta|V| + \sum_{l=1}^{|V|} \sum_{d_i \in c_k} d_{il}} . \tag{4}$$

In Laplace method, δ is set to 1. In ELE and "add-tiny" [8], δ is set to 0.5 and $1/|V|$ respectively. Add- λ smoothing determines δ by cross validation [12].

Good-Turing Smoothing. Let r represent the frequency of a given feature, N_r is the number of features with a frequency of r , $N = \sum r * N_r$ is the total number of features observed, and the value r^* is the estimated frequency. A precise statement of the theorem underlying the GT method is that

$$r^* = (r + 1) * \frac{N_{r+1}}{N_r} . \tag{5}$$

The total probability of the unseen features is N_1/N , and the probability of each feature that occurred at least once is r^*/N . LGT estimate and SGT estimate are two improved versions of GT method [8].

3 Removing Smoothing from Naive Bayes Text Classifier

Because smoothing suffers many negative effects, we propose two strategies that not only make NB avoid smoothing without sacrificing categorization accuracy, but also include the OOV features.

3.1 NB_TF Strategy

If we can make sure each feature appearing in the test document d_i also appears in all classes, smoothing is not required in the categorization procedure of d_i .

NB_TF strategy is described in pseudocode as follows:

For each test document d_i belonging to test set T . {

1. For each training document d_x belonging to class c_k , we get a $|C|$ -dimension vector $b_x = (b_{x1}, \dots, b_{x|C|})$, the k -th element $b_{xk} = 1$, others are zero.
2. For test document d_i , we get a $|C|$ -dimension vector $b_i = (b_{i1}, \dots, b_{i|C|})$,in which $b_{ik} = P(c_k)(k = 1, \dots, |C|)$, and $P(c_k)$ is computed as (2).
3. We use the training set D and the test document d_i to build the model, and call the document set E . The two quantities required in (1) are computed as follows.

$$P(c_k) = \frac{\sum_{d_x \in E} b_{xk}}{|E|} . \tag{6}$$

$$P(t_j|c_k) = \frac{\sum_{d_x \in E} d_{xj} * b_{xk}}{\sum_{l=1}^{|V^*|} \sum_{d_x \in E} d_{xl} * b_{xk}} . \tag{7}$$

in which $|V^*|$ is the vocabulary size after considering the OOV features in test document d_i .

4. We use (1),(3),(6),(7) to predict the class of test document d_i .

}

3.2 NB-TS Strategy

NB-TF which does well in online categorization is too time costly to be applied in batch categorization. In this section, we describe a new strategy that can do the batch categorization very efficiently and effectively. The new strategy, NB-TS, is described in pseudocode as follows:

1. The same as step 1 in NB-TF described in Sect. 3.1.
2. For each test document d_i belonging to test set T , we get a $|C|$ -dimension vector $b_i = (b_{i1}, \dots, b_{i|C|})$, in which $b_{ik} = P(c_k)(k = 1, \dots, |C|)$, and $P(c_k)$ is computed as (2).
3. We use the training set D and the test set T to build the model, and call the document set E . The two quantities required in (1) are computed as (6),(7).
4. We use (1),(3),(6),(7) to predict the class of each test document d_i in T .

4 Experiments and Analysis

We carry out experiments on mini-newsgroups corpus [13] over several settings and use the three-fold cross validation method to evaluate the effectiveness of Laplace smoothing, SGT smoothing, NB-TF and NB-TS. The experimental results are listed in Table 1, each bold figure in the table presents the winner of algorithms in one comparison. We explain the four settings as follows:

- Setting 1: tokenization, stopwords removal, stemming.
- Setting 2: tokenization, stopwords removal, stemming, length normalization.
- Setting 3: tokenization, stopwords removal, stemming, feature selection.
- Setting 4: tokenization, stopwords removal, stemming, feature selection, length normalization.

Precision, recall and $F_1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$ are standard effectiveness measures used widely in TC [1], and we introduce all of them in comparisons.

SGT smoothing is not suitable for Setting 2 and Setting 4 because it requires the frequency of each feature be integer, but after length normalization the frequency becomes decimal, which limits the application of SGT. By comparison, we find NB-TS performs the best in all settings without exception. NB-TF does better than Laplace smoothing in three settings out of four, except in Setting 3. SGT smoothing performs well in Setting 1 where it beats Laplace smoothing and NB-TF, but ranks last in Setting 3, in which Laplace smoothing gets its only victory over SGT smoothing and NB-TF.

Table 1. Experimental results on mini_newsgroups corpus

Settings	Methods	Micro-Values	Macro-Precision	Macro-Recall	Macro- F_1
Setting 1	NB_L	0.6966	0.7392	0.6970	0.6944
	NB_SGT	0.7202	0.7450	0.7205	0.7173
	NB_TF	0.7069	0.7389	0.7073	0.7072
	NB_TS	0.7467	0.7627	0.7470	0.7458
Setting 2	NB_L	0.7393	0.7581	0.7396	0.7380
	NB_TF	0.7585	0.7693	0.7587	0.7575
	NB_TS	0.7717	0.7808	0.7720	0.7701
Setting 3	NB_L	0.7275	0.7375	0.7279	0.7246
	NB_SGT	0.7216	0.7319	0.7220	0.7171
	NB_TF	0.7246	0.7372	0.7249	0.7187
	NB_TS	0.7408	0.7467	0.7411	0.7370
Setting 4	NB_L	0.7349	0.7429	0.7352	0.7318
	NB_TF	0.7452	0.7504	0.7455	0.7411
	NB_TS	0.7496	0.7521	0.7499	0.7449

We perform another set of experiments on the ModApte [14] split of Reuters-21578 collection [1,2]. After pruning documents having multiple topics, experiments are performed on the 25 most frequent categories. The experimental results are listed in Table 2. Once again NB_TS performs the best, followed by SGT smoothing and NB_TF, and Laplace smoothing takes the last rank.

Table 2. Experimental results on the Reuters-21578

Methods	Micro-Values	Macro-Precision	Macro-Recall	Macro- F_1
NB_L	0.9323	0.9015	0.7080	0.7558
NB_SGT	0.9396	0.8749	0.8081	0.8295
NB_TF	0.9343	0.8782	0.7537	0.7895
NB_TS	0.9404	0.8575	0.8413	0.8413

Laplace smoothing is a simple and efficient method, but the effectiveness is not very encouraging. SGT smoothing is a good method, but its applicability is limited in TC. Many proven effective preprocessing strategies may make it not applicable. NB_TF performs better than Laplace smoothing, the advantage is more distinct when length normalization is adopted. As a whole, NB_TS outperforms all the others. It has many advantages such as simplicity, effectiveness and robustness, and it is as efficient as Laplace smoothing.

5 Conclusions

In this paper, we propose two new strategies that make Naive Bayes classifier avoid smoothing without sacrificing categorization accuracy. NB_TF does a good job in TC, and it is suitable for online document categorization. When there

are many documents to classify, NB_TF is not efficient enough, while NB_TS performs better. From the experiments, we find NB_TS outperforms Laplace smoothing, SGT smoothing and NB_TF, and it has many advantages such as simplicity, effectiveness, efficiency and robustness. In a word, NB_TF and NB_TS strategies are suitable for Naive Bayes classifiers in TC.

NB_TF adds a single test document per time, and NB_TS adds the whole test set in a step. They are the two extremes. In future work, we will do some research on combining NB_TF and NB_TS strategies to find whether there exists an optimal point between the two extremes.

References

1. F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1-47, 2002
2. Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69-90, 1999
3. McCallum, A. and Nigam, K. A Comparison of Event Models for Naive Bayes Text Classification. in *AAAI-98 Workshop on Learning for Text Categorization*, Madison, WI, 1998, 41-48
4. Dmitry Pavlov, Ramnath Balasubramanyan, Byron Dom, Shyam Kapur, Jignashu Parikh. Document preprocessing for naive Bayes classification and clustering with mixture of multinomials. *KDD 2004*: 829-834
5. J. Rennie, L. Shih, J. Teevan, and D. Karger. Tackling the poor assumptions of naive Bayes text classifiers. In *ICML*, 2003
6. S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical report, Center for Research in Computing Technology, Harvard University, Cambridge, USA, 1998
7. Wang, Yong, Julia Hodges, and Bo Tang. Classification of Web documents using a naive Bayes method. *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, Sacramento, CA., November 3-5, 2003
8. W. Gale. Good-Turing Smoothing Without Tears. In *Journal of Quantitative Linguistics*, 2:217-37, 1995
9. ChengXiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, Vol. 2, Issue 2, April 2004, pp 179-214
10. Good, I. The population frequencies of species and the estimation of population parameters. *Biometrika*, v. 40, pp. 237-264, 1953
11. Fuchun Peng, Dale Schuurmans. Combining Naive Bayes and n-Gram Language Models for Text Classification. *ECIR 2003*: 335-350
12. T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. In *Information Retrieval*, volume 4, pages 5-31, 2001
13. <http://www-2.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes/mini-news-groups.tar.gz>
14. <http://www.cs.ualberta.ca/~bergsma/650/>

Medical Image Clustering with Domain Knowledge Constraint

Pan Haiwei, Jianzhong Li, and Zhang Wei

Dept. of Computer Science, Harbin Institute of Technology, Harbin, P. R. China
heaven_007cn@yahoo.com.cn
{lijzh, wzhang74}@hit.edu.cn

Abstract. Clustering medical images is an important part in domain-specific application image mining because there are several technical aspects which make this problem challenging. In this paper, we firstly quantify the domain knowledge about brain image, and then incorporate this quantified measurement into the clustering algorithm. Our algorithm contains two parts: (1) clustering regions of interest (ROI) detected from brain image; (2) clustering images based on the similarity of ROI. We apply the method to cluster brain images and present results to demonstrate its usefulness and effectiveness.

1 Introduction

Advances in image acquisition and storage technology have led to tremendous growth in very large and detailed image databases [3]. A vast amount of image data is generated in our daily life and each field, such as medical image, etc. These images involve a great number of useful and implicit information that is difficult for users to discover.

Image mining can automatically discover these implicit information and patterns from the high volume of images and is rapidly gaining attention in the field of data mining. Image mining is more than just an extension of data mining to image domain. It is an interdisciplinary endeavor that draws upon image processing, image retrieval, machine learning, database and data mining, etc. While some of individual fields in themselves may be quite matured, image mining, to date, is just a growing research focus and is still at an experimental stage. Research in image mining can be broadly classified to two main directions: (1) domain-specific applications; (2) general applications [4]. Data mining in medical images belongs to the first direction.

A few interesting studies and successful applications involving image mining have been reported. For example, the SKICAT system [1] integrates techniques for image processing and data classification in order to identify “sky objects” captured in a satellite picture set. A multimedia data mining system prototype MultiMediaMiner [2,3] uses a data cube structure for mining characteristic, association, and classification rules. In [6], localization of the visual features, their spatial relationships and their motion in time (for video) are presented. A discovering association rules algorithm based on image content from a simple image dataset is presented in [7]. Some algorithms used in the medical images [5, 8] are generally for classification. [9] proposes the clustering methods in relational databases. They are not suitable to cluster the medical images because there are important differences between relational databases

versus image databases: (1) Absolute versus relative values. In relational databases, the data values are semantically meaningful. However, in medical image databases, the data values themselves may not be significant unless the domain of medicine supports them. (2) Spatial information (Independent versus dependent position).

In this paper, we firstly use water immersion method to automatically detect ROIs and quantify the domain knowledge about brain image and ROIs to form the features, and then incorporate this quantified measurement into the clustering algorithm. Our algorithm contains two parts: (1) clustering regions of interest (ROI) detected from brain image; (2) clustering images based on the similarity of ROI.

The rest of the paper is organized as follows: section 2 is pre-processing. Medical Image Clustering is presented in section 3. Section 4 briefly reports our performance study. Conclusion is presented in section 5.

2 Pre-processing

We used CT scan images because this modality is the most used in radiotherapy planning for two main reasons. The first reason is that scanner images contain anatomical information which offers the possibility to plan the direction and the entry points of the radiotherapy rays which have to target the tumor and to avoid some risk organs. The second reason is that CT scan images are obtained using rays, which is the same physical principle as radiotherapy. This is very important because the radiotherapy rays intensity can be computed from the scanner image intensities.

In this paper, we only use water immersion method to detect ROI in medical images, see figure 1, then we combine these ROIs with their location, size and other descriptors to form a table for mining.

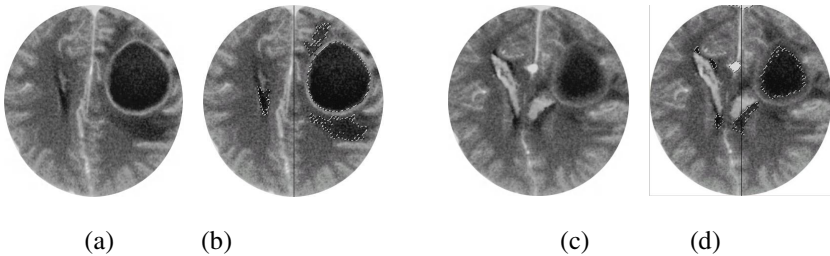


Fig. 1. Figure (a) and (c) are two original abnormal brain images. Progressive water immersion algorithm is used to mark the objects with dotted line in figure (b) and (d).

3 Medical Image Clustering

By applying water immersion algorithm, we segment images into objects. Let $IM = \{IM_1, IM_2, \dots, IM_N\}$ be a image set. After the above algorithm, Each image IM_j contains k objects $R_{j1}, R_{j2}, \dots, R_{jk}$. For different image, k may be not equal. Let the total of objects in IM be M , then we denote the object set as $R = \{R_1, R_2, \dots, R_M\}$.

3.1 Feature Extraction

The domain knowledge of the brain image characteristics indicates that the normal persons have nearly the same brain structure that is evident to be bilateral symmetry, see figure 2. The pathological regions result in irregularly shaped grey level distribution in the CT scan images and destroy the symmetry, see figure 1. At this point, the following relevant features are able to provide sufficient discriminative power to cluster objects into different groups: (1) grey level of the object of interest; (2) area of the object of interest; (3) location of the object of interest; (4) elongation of the object of interest; (5) direction of the object of interest; and (6) symmetry of ROI.



Brain midline

Fig. 2. Normal person's brain image

With the above method, the object extracted from brain image is either brightness or darkness. So we define grey level as $GL=0$ (brightness), or 1 (darkness). The second feature relate to the size of the region. Area of the region of interest is defined as the total number of pixels within the region, up to and including the boundary pixels. The location of ROI is simply defined with the coordinates of the centroid:

$$\bar{x} = \frac{\iint x\delta(x, y)dA}{k} \quad \bar{y} = \frac{\iint y\delta(x, y)dA}{k} \tag{1}$$

$\delta(x, y)$ is surface density function and k is the number of the pixels of the object.

The fourth feature is the elongation of an object which is defined as the ratio of the width of the minor axis to the length of the major axis. This ratio is computed as the minor axis width distance divided by the major axis length distance, giving a value between 0 and 1.

$$elongation = len_{major} / len_{minor} \tag{2}$$

where len_{major} is the major axis length of the object and len_{minor} is the minor axis length of the object. We establish a common coordinate with brain midline as y and perpendicular line going through the midpoint of midline as x . With the major axis of the object, we define the next feature, direction of the object, as the inclination θ of the major axis and x positive direction and $\theta \in [0, 2\pi)$.

Before introducing the final feature, we will give some definitions.

Definition 3.1. Pixel set of IM_p is defined as $P=\{p_i|p_i$ is the pixel with coordinate (x_i, y_i) in image $IM_p\}$, $P(L)$ and $P(R)$ are pixel set of $IM_p(L)$ and $IM_p(R)$ respectively. For any $p_{li} \in P(L)$, $p_{ri} \in P(R)$, they are symmetric pixel if the line between p_{li} and p_{ri} is halved vertically by brain midline. They are denoted as p_{li} and p_{ri} .

Definition 3.2. $\Delta g(P)$ is IM_p 's difference set if for any symmetrical pixel p_{li} and p_{ri} , $\Delta g(P)=\{ \Delta g_i|\Delta g_i=g(p_{li})-g(p_{ri}), i=1,2,\dots,|p|/2 \}$, where $g(p_{li})$ and $g(p_{ri})$ are grey level of these two pixels respectively.

Now we define the sixth feature as following.

Definition 3.3. An object of interest R_{pj} is symmetrical if for IM_p and $R_{p1}, R_{p2}, \dots, R_{pk}$, $P'=P - \sum_{i=1}^k P(R_{pi})$, the mean grey level of $\Delta g(R_{pj})$, $mean(\Delta g(R_{pj}))$, and the mean grey level of $\Delta g(P')$, $mean(\Delta g(P'))$, satisfy the following condition:

$$|mean(\Delta g(R_{pj})) - mean(\Delta g(P'))| < \epsilon.$$

Theorem 1. If for IM_p , an object of interest R_{pk} is symmetrical, then there must exist another object of interest R_{pt} that satisfies the following conditions: (1) R_{pk} and R_{pt} must lie in different side of the midline; (2) R_{pk} and R_{pt} are either two different objects or the same one.

3.2 Object Clustering

Now, we would like to determine similarity between two objects based on these features. For an object R_i , a vector $V_i(v_{i,1}, v_{i,2}, \dots, v_{i,k})$ is constructed and similarity between object R_i and object R_j is as follows:

$$Sim^R(R_i, R_j) = \Delta v_{ij,1} * (\sum_{h=2}^{k-1} v_{i,h} * v_{j,h}) / (\sqrt{\sum_{h=2}^{k-1} v_{i,h}^2} * \sqrt{\sum_{h=2}^{k-1} v_{j,h}^2}) \tag{3}$$

According to domain knowledge, we let $v_{ij,1}$ be GL. Two objects are not possible to be similar if one is very darkness and the other is very brightness. So we let $\Delta v_{ij,1}$ be 1 if $v_{i,1} = v_{j,1}$, or 0 if $v_{i,1} \neq v_{j,1}$ in formula (3). Another similar rule is that if two objects R_i and R_j satisfy the second condition in theorem 1, they will be grouped into the same cluster. This rule is prior to the above similarity function.

To construct object clusters we use a threshold T_R . If similarity between two objects is smaller than T_R , then the two objects can be in the same group. We define ϵ -adjacent area (ϵ -AA) as the object set in which similarity between each pair of objects in the same group must be smaller than T_R .

Object Clustering Algorithm (OCA):

Input: object set R, T_R and MP

Output: p clusters

1. Assume that the size of object set R is M and examine ϵ -AA of these M objects;
2. If (ϵ -AA of R_i involves more than MP objects)
3. Then mark R_i as the core object;
4. While (all core objects) {
5. Clustering all density reachable objects; }

Time complexity of this algorithm is $O(n \log n)$.

3.3 Image Clustering

The OCA algorithm clusters these M objects into p groups which we denoted as $RC = \{C_1 \square C_2 \square \dots \square C_p\}$. Image clustering is based on image similarity. To calculate image similarity, we construct a vector $W_i(w_{1,i}, w_{2,i}, \dots, w_{p,i})$ for an image IM_i . $w_{p,i}$ is the weight of object cluster C_p in image i. In this vector we keep the weight of each group. Thus, the size of vector W_i is same as the total number of object clusters ($=p$). It is possible that the weight of a group may be zero. This is because no object of an image may be a participant of that group during clustering.

Definition 3.4. Let $CIM_{j,i}$ be the times of objects of cluster C_i in image IM_i , IMC_j be the number of images in which objects of cluster C_i appear. We define

$$iIMC_j = \log(N/IMC_j) \tag{4}$$

where N is the size of the image set. For a vector $W_i(w_{1,i}, w_{2,i}, \dots, w_{p,i})$, we define

$$W_{j,i} = CIM_{j,i} * iIMC_j \tag{5}$$

After computing image vectors, we get similarity between any two images using cosine similarity:

$$\text{Sim}^{\text{IM}}(\text{IM}_i, \text{IM}_j) = \left(\sum_{h=1}^p w_{h,i} * w_{h,j} \right) / \left(\sqrt{\sum_{h=1}^p w_{h,i}^2} * \sqrt{\sum_{h=1}^p w_{h,j}^2} \right) \tag{6}$$

To cluster images, we also use a threshold T_{IM} . If similarity between two images is smaller than T_{R} , then the two images can be in the same group. The algorithm will stop when the images are clustered into k group.

Image Clustering Algorithm (ICA):

Input: Image set IM and the number of clusters k ;

Output: k clusters

1. Each element of IM is regarded as an atomic cluster and compute $\text{Sim}^{\text{IM}}(\text{IM}_i, \text{IM}_j)$;
2. Clustering in terms of similarity;
3. While (the number of clusters is not equal to k) {
4. If all R_i s in one image are symmetrical
5. Then cluster this image to *special* cluster;
6. Compute Sim^{IM} ;
7. Clustering the sub-clusters or images; }

Time complexity of this algorithm for the worst case is $O(n^2)$.

4 Experiments

The dataset utilized in our experiments was real data from hospital. The main reason why we study on real brain CT images instead of any simulative data is to avoid insignificance and uninterestingness and the reliability of the discovered knowledge. To have access to real medical images is a very difficult undertaking due to legally privacy issues and management of hospital. But with some specialists' help and support, we got 618 precious images and their corresponding diagnosis records which, for simplicity, we generalized to normal(N) and abnormal(A).

To measure the quality of a cluster, we use precision and recall. Recall is the ratio of relevant images to total images for a given category. Precision is the ratio of relevant images to images that appear in a cluster for a given category. In figure 3, x axis represents different number of clusters and the y axis represents p and r . We have observed that precision and recall are higher for medical images. Figure 4 shows an instance of our clustering algorithm. In any case fairly large medical data sets exist but they are not available to us. Also, it would be interesting to apply these ideas in other domains where large complex data sets are available.

5 Conclusion

In this paper, we have used a water immersion method to preprocessing the medical image set to detect objects in medical images. Then we proposed two new algorithms with guidance of domain knowledge to cluster the medical images. We quantified the

domain knowledge and use them in the clustering algorithm. We have described the problem with a general form to provide a common framework for other problems appeared in other domains.

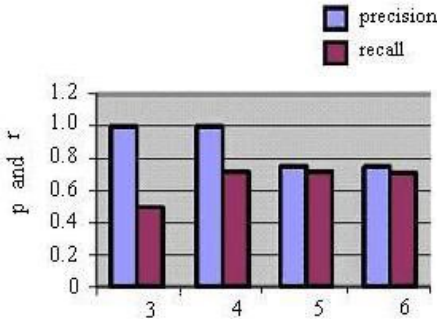


Fig 3. Cluster quality for different number of clusters

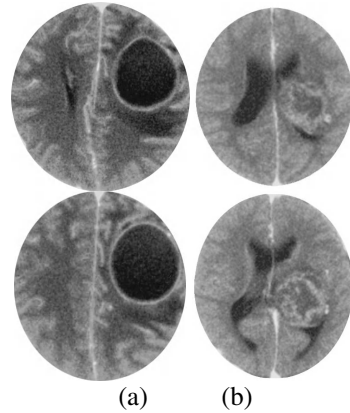


Fig 4. (a) and (b) show two images of two different clusters

References

- [1] U. M. Fayyad, S. G. Djorgovski, and N. Weir. Automating the analysis and cataloging of sky surveys. In U. Fayyad, G. Pietetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 471–493. AAAI/MIT Press, 1996.
- [2] O. R. Zaiane, J. Han, Z.-N. Li, J. Y. Chiang, and S. Chee. MultiMediaMiner: A system prototype for multimedia data mining. In *Proc. ACM-SIGMOD*, Seattle, 1998.
- [3] O. R. Zaiane, J. Han, Z.-N. Li, and J. Hou. Mining multimedia data. In *CASCON'98: Meeting of Minds*, Toronto, 1998.
- [4] WYNNE HSU, MONG LI LEE, JI ZHANG. Image Mining: Trends and Developments. *Journal of Intelligent Information Systems*, 19:1, 7–23, 2002.
- [5] Wynne Hsu, Mong Li Lee, Kheng Guan Goh. Image Mining in IRIS: Integrated Retinal Information System. *Proceedings of the ACM SIGMOD*, May 2000, Dallas, Texas, U.S.A., pp. 593.
- [6] Osmar R. Zaiane, Jiawei Han, Hua Zhu. Mining Recurrent Items in Multimedia with Progressive Resolution Refinement. *ICDE 2001*.
- [7] Ordonez, C. and Omiecinski, E. (1999). Discovering Association Rules Based on Image Content. In *IEEE Advances in Digital Libraries Conference*.
- [8] Osmar R. Zaiane, Maria-Luiza Antonie, Alexandru Coman. Mammography Classification by an Association Rule-based Classifier. *Proceedings of the Third International Workshop on Multimedia Data Mining (MDM/KDD'2002)*.
- [9] Christian Bohm, Karin Kailing, Peer Kroger, Arthur Zimek. Computing Clusters of Correlation Connected Objects. In *Proc. 2004 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'04)*.

Tick Scheduling: A Deadline Based Optimal Task Scheduling Approach for Real-Time Data Stream Systems¹

Zhengyu Ou, Ge Yu, Yaxin Yu, Shanshan Wu, Xiaochun Yang, and Qingxu Deng

Dept. of Computer Science and Engineering, Northeastern University, P.R. China
xiaoyuzhou1998@163.com

Abstract. Most of the current research work on timely streaming data processing focuses on minimizing average tuple latency instead of strict individual tuple latency upper-bound, that is, deadline. In this paper, we propose a novel deadline-scheduling strategy, namely tick scheduling (TS), dealing with applications with specified deadline constraints over high volume, possibly bursting, and continuous data streams. We demonstrate that TS policy, which combines precise batch scheduling plan construction and adaptive batch maintenance mechanism can significantly improve system performance by greatly reducing system overheads and adapting gracefully to the time-varying data arrival-rate. Experimental results show the significant improvements provided by our proposed policy.

1 Introduction

With the popularity of real-time data stream processing in many mission-critical applications, such as traffic monitoring, production controlling, military commanding, and various sensor networks, it is required to process multiple, high volume, rapid, possibly bursting, continuously arriving data streams for the queries with specified deadline constraints. In other words, each streaming data must be processed within the deadline of the corresponding query; otherwise the processing on the data is worthless. In this paper, we focus on deadline scheduling problem dealing with such applications.

Real-time scheduling problem over data streams has motivated a considerable research recently [1,2,3,4]. One major class is batching strategies [1,2], which, usually based on statistic computing (such as estimating the operator selectivity, scheduling cost etc.), are impractical to the mission-critical applications requiring precise deadline. Another major class is tuple-at-a-time strategy, as exhibit great adaptivity in query processing [3,4]. However, those approaches will introduce great scheduling overheads, especially when dealing with bursty streams.

¹ This work is supported by National Science Foundation (60473073) and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of the Ministry of Education.

Besides, scheduling strategies [5,6,7] proposed in real-time database systems attempt to satisfy deadlines associates with each incoming task, with the goal of minimizing the number of tasks that miss their deadlines. These strategies commonly based on task at a time; thus, they also have the deficiency of high system overheads.

In this paper, we focus on precise batching. Moreover, it is very crucial for scheduling plans to adapt to time-varying data arrival-rate. In the TS strategy, an adaptive batch maintenance mechanism is deployed for such adaptivity.

The rest of the paper is organized as follows. Section 2 presents the preliminary. Section 3 proposes the deadline scheduling strategies over data stream. Section 4 shows the performance experiments. Section 5 describes the related works. Finally, the conclusion is given in Section 6.

2 Preliminary

In this section, we first describe the Real-time data stream tasks. Then give the definition of *tick*. Finally, we present the performance metric.

Real-time data stream tasks. In a real-time data stream system, each tuple in a data stream can be represented by $\langle r, \tau \rangle$, where r is the data and τ is the timestamp denoting the time the data is generated by the data source. We define deadline over data stream as follows.

Definition 1 (deadline). For a tuple $\langle r, \tau \rangle$ processed by query Q , the latency upper bound is U_Q , the deadline of the tuple processed by query Q is $\tau_r + U_Q$.

Here, a tuple processed by a query Q is called to be a task. We assume each query has an input tuple queue, hence, each query corresponds to a task queue. Moreover, in a DSMS, queries are continuous where a query is executed whenever new data arrives. Hence, a continuous query is actually corresponds to a continuous task stream. And the purpose of a real-time data stream system for mission-critical applications is to process as many tasks as possible within the deadlines.

Definition of tick. Here, we assume ε is the measure of the timestamp. Due to the rapid speed of data stream, different data may be generated in the same time interval ε , and thus have the same timestamp. According to definition 1, the corresponding tuples will have the same deadline. Deadline also have a measure, we denote it as *tick*. Here, we have $tick \geq \varepsilon$ (If $tick \leq \varepsilon$, different data have the same timestamp may arrive in the different time interval of U_Q , and thus should have different deadline. However, according to $d = \tau + T_i$, they actually have the same deadline.). As a result, for a certain query, the data generated in the same time interval *tick* will have the same query. We call the data generated in the same interval a tick batch and partition each query's input queue into tick batches. And the generated batch queues of each query are called tick queues. hence, each tick batch in the tick queues correspond to a batch of tasks.

Performance Metric. The performance of scheduling strategies is typically measured by *deadline-missed ratio (DMR)* in traditional real-time systems. The *DMR* is defined as the ratio: the number of tasks missed their deadlines to the total number of the tasks. And the total number of the tasks is usually a known constant. However,

as we discussed above, an unbounded continuous query corresponds to an unbounded task stream. Therefore we will define *DMR* over stream as follows:

Definition 2 (DMR). $DMR(Q_i) = \Delta N_i / N_i$, where $DMR(Q_i)$ is the *deadline-missed ratio* of query Q_i , ΔN_i is the number of the tuples can not be processed by query Q_i within the deadlines, and N is the total number of tuples that have arrived at the tuple queue of query Q_i . Accordingly, the *deadline-missed ratio* in a DSMS is $\Sigma(DMR(Q_i)*w_i) / \Sigma w_i$, where w_i is the magnitude of query Q_i , the more important the query, the larger w_i .

We observe that the more tuples processed by a query Q , the less $DMR(Q)$. As a result, the less DMR , the better the system performance.

3 Tick Scheduling (TS)

Figure 1 illustrates the architecture of TS scheduler, which includes mainly four components: *tick queues*, tick monitor, *adaptive maintenance*, and *TBS (tick-based batch scheduling) scheduler*.

We now describe the *tick*-based batch scheduling (TBS) strategy designed in TBS scheduler.

3.1 TBS Strategy

As we can see from Figure 1, TBS scheduler include three components: *batch task queues*, *DMR*, *priority computation*. However, statistic computation and load shedder are also components of TBS scheduler.

We do not display the components in Figure 1 in that they would bring great deficiency to system performance and are excluded by the scheduler. As we discussed above, tuples in the same tick of the tick queues has the same deadline. We timestamp every ticks in the tick queues with their deadlines and obtain batch task queues. The main steps of the *tick*-based batch-scheduling algorithm can be described as follows.

Step1: Call the priority computation to compute the highest priority batch task.

Step2: Estimate (using the operator selectivity, scheduling cost etc.) how many tasks in the batch task could be processed before the deadline and sample the batch task accordingly.

Step3: Send the batch task to the query engine.

Step4: After the batch task is processed, modify *DMR*, delete the batch task and inform the tick queues.

Step5: If there exists outdated batch tasks in the batch task queues, delete them from the batch queues and inform the tick queues.

Step6: If new ticks insert the tick queues, modify the batch task queues.

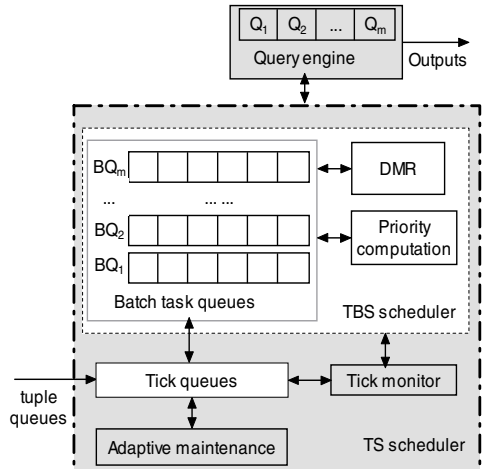


Fig. 1. Architecture for TS scheduler

Here, the priority computation is based on EDF [5]. Compared to the common task at a time strategies, TBS strategy, a basic batching method, can significantly reduces the system overheads such as scheduling overhead (including maintenance of the execution queue and memory management), calls to the query plan, and context switch, as results in sharply decrease in *DMR*. However, as we have discussed in Section 1, statistic computation is ill suit to precise deadline scheduling. Hence, TBS* strategy is designed to address the problem.

3.2 TBS* Strategy

Definition 3 (critical point). When processing the k -th task in a batch task with N tasks, the deadline comes. We call the k is the critical point of the batch task.

Hence, it is necessary for a deadline-scheduling strategy to implement critical point scheduling (in other words, process exactly the tasks in the batch task before the critical point and reject the latter ones.). We now describe how tick monitor do this job. We observe that *tick* is a time interval. Assume it is a second. Hence, we only need to check the currently existing batch tasks at the very beginning of every second. If a batch task does not miss its deadline at the very beginning of a second, it will not miss the deadline during the second. At the very beginning of every tick, tick monitor will be activated. First, the tick monitor interrupts the current executing batch task, and then calls priority computing. Finally, return to the TBS algorithm. Here, TBS* scheduler is the combination of tick monitor and TBS scheduler.

Deploying tick monitor mechanism, TBS* scheduler can implement precise batching, as will greatly cutt down the *DMR*.

3.3 TS Strategy

If there is no batch task in the batch task queues, the BTS* scheduler will wait until all the tasks in the coming batch task arrives. In other words, although there may exist tasks in the batch task queues, the BTS* scheduler still keep wait. As a result, the strict system resources are wasted, especially when dealing with time-varying, bursty data streams. We propose adaptive maintenance mechanism to solve the problem. Here, TS scheduler is the combination of adaptive maintenance and BTS scheduler. The main steps of adaptive maintenance are as follows: Take the arrived tasks in each batch task queue as whole batch tasks (to make difference, namely fragment batch task). And then call BTS* algorithm to process the fragment batch tasks. Loop until there exist batch tasks in the batch task queues or the tick monitor is activated. As a result, TS strategy can make full use of strict system resources, and shows great adaptivity to the changing characteristics of data streams.

Table 1. Simulation of Parameters

Parameter	Value
query number	40
Query depth	1-3
Query fan-out	1-6
operator cost	1-20us
slectivity	0.01-1
overhead cost	50us
inter arrival time skewness	0 - 0.9
query deadline	1.0-5.0s
<i>tick</i>	0.1s

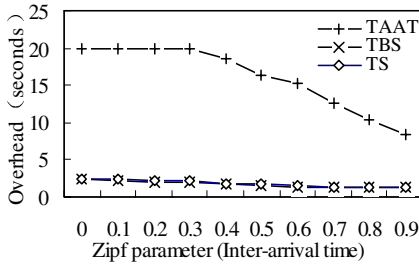


Fig. 2. Overheads under skewness of inter-arrival time

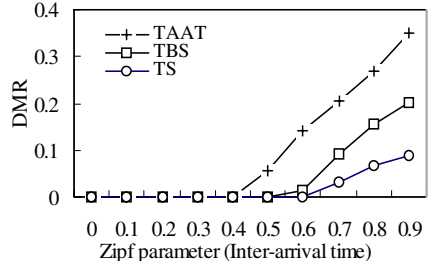


Fig. 3. Static composing

4 Experiments

In this section, we propose experimental results to compare the performance between the deadline scheduling strategies over data streams including task at a time (TAAT) policy, TBS, TS. Due to lack of space, we only give the experiments that highlight the efficiency and effectiveness of the TS strategy.

4.1 Experimental Setup and the Implementing Details

In our experimental evaluation, we simulated the execution of a continuous query as in [9]. Where the query is specified by two parameters: depth and fan-out. The depth of the query specifies the number of levels in the query tree and the fan-out specifies the number of children for each operator. Each leaf operator is attached to a data stream. The inter-arrival time of tuples is generated according to a Zipf distribution to simulate the time varying, bursting natures over data streams. The stream length is 10K tuples. Table 1 summarizes all our simulation parameters. We implement our algorithm on top of Debian Linux using C++. All the experiments were run on an Intel machine with P4 2.0GHz CPU, 512MB main memory.

4.2 Performance Evaluation

We first investigate the system overheads introduced by TAAT, TBS and TS, as is shown in Figure 2. We observe that TAAT is significantly worse than the other two strategies, while TBS and TS shows great efficiency in reducing the overheads, and their system overheads are almost the same. The reason is that, both TBS and TS introduce *tick*-based batch processing mechanism, which benefits a lot to reduce the system overhead. Besides, the overhead of TS is a bit more than TBS owing to the additional use of TM in TS, whereas the TM does not make for the key components of scheduling overhead.

Additionally, we notice that as the skewness of the Zipf parameter increases, the overheads of the three methods may decrease accordingly. This is because the average tuple arrival rate increases together with the increasing skewness. And when the rate exceeds the capacity of the system, the tasks missing the deadlines are dropped. While the total number of the tasks is a constant (due to constant stream length), accordingly, the overhead over the tasks meeting their deadline will decrease. Hence,

before the skewness increases to 0.4, the TAAT has underlined overheads in that the increasing rate does not saturate system load unless the skewness is larger than 0.4.

Essentially, Figure 3 shows the same behavior illustrated in the previous experiment. That is, compared to TAAT, BST and TS perform particularly better in cutting down *DMR*. Besides, the first thing to notice is that TS has better *DMR* than TBS. On account of statistic computing, TBS cannot achieve precise batch processing, which contributes more to the *DMR* than their difference of the overhead does. Another interesting feature of the curves in the Figure 3 is the location of the first non-zero point. The first non-zero point indicates the saturation caused by the policy. Hence, TBS and TS policies are more adaptive to high load.

5 Conclusions

The continuous, bursty nature of streaming data and timely requirements of continuous queries pose great challenges to the existing data processing techniques. In this paper, we address the fixed time scheduling problem over data streams.

We show that the naïve approach based on task at a time does not scale due to the high system overheads. We first produce a basic batch scheduling strategy, TBS, which significantly cut down the system overheads. Subsequently, we propose TS strategy to overcome the deficiency of TBS strategy induced by statistic computing. For the goal of adapting to the time varying data arrival-rate, we further improve TS strategy, which results in full usage of strict system resources. Our extensive experimental evaluation shows the significant improvements provided by our proposed TS policy.

References

1. Carney D., Centintemel U., et al. Operator scheduling in a data stream manager. Proc. of the 29th VLDB Conf., Berlin, Germany, Sept. 2003, pp.838-849.
2. Babcock B, Babu S, Datar M, Motwani R. Chain: Operator scheduling for memory minimization in data stream systems. Proc. of SIGMOD Conf., San Diego, Jun.2003, pp. 253-264.
3. Avnur R., Hellerstein J., Eddies: continuously adaptive query processing[A]. Proc. of the SIGMOD Conf., Dallas, USA, Jun. 2000.
4. Chandrasekaran S. and Franklin M., PSoup: a system for stream queries over streaming data[J], The VLDB Journal, Dec.2003. pp.140-156.
5. C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in hard real-time traffic environment", *Journal of the Association for Computing Machinery*, vol. 20, no. 1, Jan. 1973.
6. Stewart D B. Real-time software design and analysis of reconfigurable multi-sensor based systems [Ph D dissertation], Carnegie Mellon University, Pittsburgh, 1994.
7. Duda, K., and Cheriton, D. 1999. Borrowed-virtual-time (bvt) scheduling: supporting latency-sensitive threads in a general-purpose scheduler. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles*.

A Novel Ranking Strategy in Hybrid Peer-to-Peer Networks

Qian Zhang, Zheng Liu, Xia Zhang, Xuezhi Wen, and Yu Sun

National Engineering Research Center for Computer Software,
Northeastern University, Shenyang 110004, China
zhangqian@neusoft.com

Abstract. The state of the art of hybrid P2P-based information retrieval is still at its infant stage and confronted with many challenges. One of the most urgent problems is how to combine the retrieval results from different neighboring nodes into a single, integrated ranked list. In this paper, we propose a result merging algorithm to address the challenge. Our algorithm is deterministic and doesn't require neighboring nodes to provide any information for result merging, which makes it different from other algorithms that require cooperation from neighboring nodes. A variety of experiments demonstrate that the new approach is effective.

1 Introduction

Hybrid P2P architectures include two types of nodes. There are *leaf nodes* that provide information as well as post requests. Leaf nodes can be used to model an information resource. There are also *directory nodes* that don't have content of their own but which provide regional directory services construct and collectively work to cover the whole network. Figure 1 illustrates the hybrid P2P networks. The black nodes are directory nodes, and the gray nodes are leaf nodes.

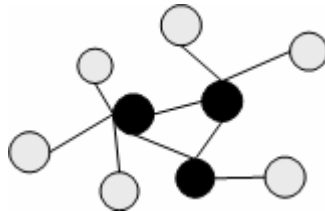


Fig. 1. Hybrid P2P Networks

In hybrid P2P networks, result merging naturally takes place at directory nodes because they provide directory services to improve the routing of information requests, which could be useful for result merging. The results that need to be merged at a directory node may include not only the results from other directory nodes down the query path, but also the results from the neighboring leaf nodes.

Result merging in hybrid P2P networks is not a simple adaptation of existing approaches due to unique characteristics of P2P environments, for example, skewed collection statistics, and higher communication costs. About the related work, in PlanetP[1], it first ranks peers according to the goodness of each peer's to the submitted query and then ranks the documents returned by these peers. However, its schema is heuristic but not deterministic and requires cooperation from peers. In[2], Zhiguo Lu have proposed a deterministic strategy, but it requires full cooperation from participants. [3] also presents a deterministic strategy, but it is just suitable for pure structured P2P systems and need global statistics such as the inverse document frequency, which can only be obtained in completely cooperative environments where each node share its document and corpus statistics. Jie Lu [4] adapted Semi-Supervised Learning (SSL) result merging algorithm to hybrid P2P networks with multiple directory nodes. A directory node that uses SSL learns a query-dependent score normalizing function for each of its neighboring nodes. This function transforms neighbor-specific documents to directory-specific document scores. However, the modified SSL algorithm requires downloading documents from neighboring nodes to server as training data that means higher communication costs. Jie Lu also proposed another result merging algorithm, which is called SESS. SESS is a cooperative algorithm that requires neighboring nodes to provide statistics information for each of their top-ranked documents.

In short, all algorithms mentioned above require cooperation from participants with the exception of modified SSL. However, SSL requires downloading documents from neighboring nodes to serve as training data that means higher communication costs [4]. In this paper, we propose a novel result merging algorithm that can be used by directory node, which is called *RMNC* (result merging with no cooperation) . *RMNC* is deterministic and doesn't require neighboring nodes to provide any statistics information, which makes it different from other algorithms that require cooperation from participants.

The remainder of the paper is organized as follows. Section 2 describes our new approach to merge results. Section 3 proposes one method for ranking the neighboring nodes based on the relevance to a given query. The scores of the neighboring nodes with respect to a given query are important for result merging. Section 4 and Section 5 explain our experimental methodology. Section 6 concludes.

2 Result Merging

Our *RMNC* algorithm is base on a function that maps local document *ranks* obtained from a neighboring node into global document *scores* which can then been merged together with document ranks from other neighboring nodes. The local document ranks mean the ranks of the documents at a neighboring node resulted from relevance scoring computed locally by the node. The global document scores mean directory-specific scores, which are the new scores of the documents after being merged. In order to explain the algorithm, we suppose a set of neighboring nodes $P = \{P_1, P_2, \dots, P_{|P|}\}$ with goodness scores $S_{1,q}, S_{2,q}, \dots, S_{|P|,q}$ has been selected for query q . Each of the neighboring nodes returns a set of documents, called result set, ranked by their relevance scores with respect to the given query q . $O_{u,v}$ denotes the rank of document

u in neighboring node v . In the final stage, K top documents are to be retrieved, where K is determined by the user.

In order to merge results from neighboring nodes, we assume that the following are good indicators: the number of relevant documents returned by each peers and its goodness scores for the submitted query. The directory-specific score is base on our intuition that the number of documents NUM_v contributed by neighboring node v to the final merged result is proportional to its goodness score. NUM_v is defined as follows.

$$NUM_v = K \times \frac{S_{v,q}}{\sum_{p=1}^{|P|} S_{p,q}} \tag{1}$$

On the other hand, we also assume that one peer node would contain more relevant documents for a given query, if it found more documents.

Based on the intuition mentioned above, we define the following local document rank to directory-specific relevance score mapping for a given query q .

$$G_{u,v,q} = K \times \frac{S_{v,q}}{O_{u,v,q} \times S_{max,q}} \times \frac{N_{v,q}}{\sum_{p=1}^{|P|} N_{p,q}} \tag{2}$$

Where $O_{u,v,q}$ is the rank of document u in neighboring node v , $G_{u,v,q}$ is the global relevance score of document u in neighboring node v , and $S_{max,q} = \max\{ S_{m,q} \mid m=1,2,\dots,|P|\}$. $N_{v,q}$ is the number of results from node v for query q . In the final stage, the resulting global relevance scores are sorted in a decreasing order of score $G_{u,v,q}$, and the best K or $K+$ documents are retrieved. The new algorithm we propose takes into account the number of relevant documents returned by each neighboring nodes. Also, our algorithm is deterministic and does not require any cooperation from participants, which means that our algorithm has many advantages compared to other result merging algorithms, for example, less communication costs.

The only information needed to compute $G_{u,v,q}$ used by Equation 2 is the relevance scores of neighboring nodes with respect to a given query q . Next we will identify the problem.

3 Neighboring Nodes Ranking

Many nodes ranking algorithms used by directory nodes require cooperation from neighboring nodes, which we would prefer to avoid. On the other hand, neighboring nodes ranking in hybrid P2P networks is not a simple adaptation of existing approaches [5-7] due to unique characteristics of P2P environments, for example, skewed collection statistics, and higher communication costs. It is a more difficult problem especially in uncooperative P2P environments. In this paper, we adapt KL [7] algorithms to rank neighboring nodes, described below.

In Xu and Croft’s language modeling framework, collections are ranked by a modification of the Kullback-Leibler divergence which is the conditional probability of predicting the collections C for a given query Q :

$$KL(Q, C) = \sum_{j=1}^{|Q|} \frac{f(Q, w_j)}{|Q|} \log \frac{f(Q, w_j)/|Q|}{(f(C, w_j) + f(Q, w_j))/(|Q| + |C|)} \quad (3)$$

Where $f(Q, w_j)$ is the term frequency of term w_j in the query, $|Q|$ is the number of term occurrences in the query, $f(C, w_j)$ is the term frequency of the term w_j in the collection, and $|C|$ is the total number of term occurrences in the collection. Kullback-Leibler divergence is a distance metric and falls in $[0; \infty]$. The smaller the value, the better the collection C predicts Q . Justification for the metric can be found in textbooks on information theory [8]. Because $f(Q, w_j)/|Q|$ is independent of any node's collection, the ranking of neighboring nodes based on $KL(Q, C)$ is equivalent to ranking based on $S(Q, C)$ scores calculated as:

$$S(Q, C) = - \sum_{j=1}^{|Q|} \log(f(C, w_j) + f(Q, w_j))/(|Q| + |C|) \quad (4)$$

The only information needed to calculate $S(Q, C)$ scores used by Equation 4 is term and term frequency information of a neighboring node's collection. As discussed above, we would prefer to avoid cooperation from neighboring nodes. So instead of obtaining the resource descriptions directly from its neighboring nodes, the directory records the query terms of past queries that the neighboring node has responded to. Next we adapt Equation 4 to ranking neighboring nodes in uncooperative P2P environments. That is:

$$S(Q, CQ_v) = \frac{1}{- \sum_{j=1}^{|Q|} \log(f(CQ_v, w_j) + f(Q, w_j))/(|Q| + |CQ_v|)} \quad (5)$$

Where CQ_v is the set of query terms that neighboring node v has responded to. Given a new query, the directory node computes scores for its neighboring nodes using Equation 5.

4 Testbed

There has been no standard data for evaluating the performance of results merging in hybrid P2P networks, so we developed one test set, which is a 6 gigabyte, 1.1 million document set downloaded from Web. The test data was divided into 2,400 collections based on document URLs. The HTML title fields of the documents were used as document names. Each of the 2,400 collections defined a leaf node in a hybrid P2P network.

Prior research shows that 85% of the queries posted at web search engines have 3 or less query terms [9], so for most documents, we only extract a few key terms as queries. Here we extract key terms from document names. Most queries had 2-3 terms and no query had more than 7 terms. 12,000 queries were randomly selected from the automatically-generated queries to be used in our experiments. For each query, a leaf node was randomly chosen to issue the query. Next we will discuss how to construct the topology of hybrid P2P network.

We use the Power-Laws method described by [10] to generate the connections between directory nodes. A directory node had on average 5 directory node neighbors and 16 leaf node neighbors. As in the Gnutella protocol [11], each message had a time-to-live (TTL) field that causes the query to expire and is not further propagated to other nodes. In our experiment, the initial TTL was set to 5 for query messages routed to directory nodes.

5 Evaluation Methodology

Top 50 documents retrieved from a single large collection were treated as the relevant documents for each query. Modified SSL algorithms are also implemented in order to have a comparison with our *RMNC* algorithm.

Recall and Precision are two classical metrics to evaluate the information retrieval technology. Here we use 11-point recall-precision to evaluate the merged retrieved results from the hybrid P2P networks.

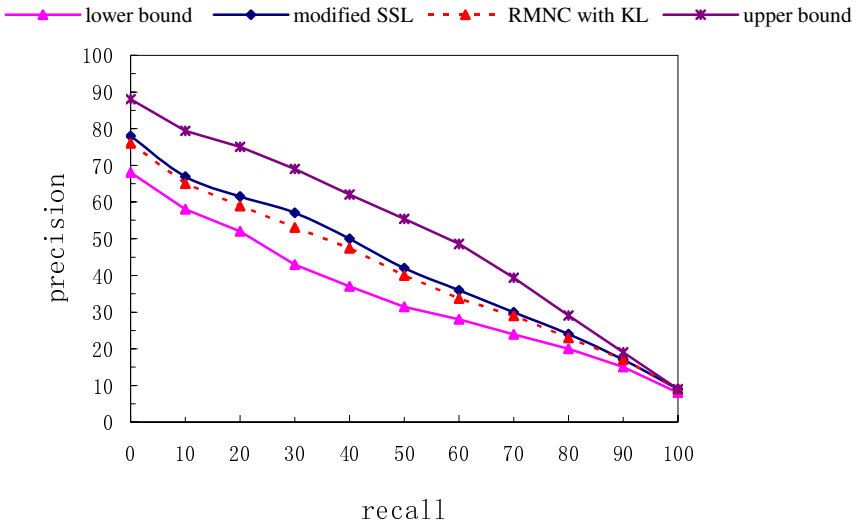


Fig. 2. The 11-Point Recall-Precision Curves for Different Result Merging Algorithms when 6,000 queries have been randomly issued

Figure 2 shows the 11-point recall-precision curves for different result merging algorithms when 6,000 queries have been randomly issued. The lower bound was generated by directly merging documents from different neighboring nodes using their local document scores. The upper bound was generated by the documents according to their corresponding scores in the retrieval results from the whole single collection. The *RMNC* algorithm using modified KL method improved the average precision by 18.1% compared with the lower bound. Our *RMNC* algorithm had near optimal performance compared with modified SSL algorithm. However, modified

SSL requires downloading large documents from neighboring nodes to serve as training data, which means higher communication costs.

6 Summary and Conclusions

In this paper, we propose a novel result merging algorithm used by directory nodes in hybrid P2P networks. Our algorithm does not require cooperation from neighboring nodes, which makes it different from other algorithms except modified SSL. However, modified SSL can induce higher communication costs. Experimental results demonstrate that our algorithm is effective and has near optimal performance compared with modified SSL.

References

1. F. M. Cuenca-Acuna, C. Peery, and R. P. M. T. D. Nguyen. Plantet: Infrastructure support for p2p information sharing. In *Technical Report DCS-TR-465*, Department of Computer Science, Rutgers University, Nov., (2001)
2. Zhiguo Lu¹, Bo Ling¹, Weining Qian¹, etc. A Distributed Ranking Strategy in Peer-to-Peer Based Information Retrieval Systems. In *Proceedings of Sixth Asia Pacific Web Conference*, China, (2004)
3. M. M. Chunqiang Tang, Zhichen Xu. Peerssearch: Efficient information retrieval in structured overlays. In *Proceedings of HotNets-1, ACM SIGCOMM*, (2002)
4. J. Lu and J. Callan. Merging retrieval results in hierarchical peer-to-peer networks (poster description). In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. (2004)
5. Callan, J.P., Lu, Z., and Croft, W. B. Searching Distributed Collections with Inference Networks. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (1995) 21-28
6. Gravano, L., and Garcia-Molina, H. Generalizing GLOSS to vector-space databases and broker hierarchies. In *Proceedings of the 21th International Conference on Very Large Databases (VLDB)*, (1995) 78-89
7. Xu, J. and Croft, W. B. Cluster-based Language Models for Distributed Retrieval. In *Proceedings of the 22th International Conference on Research and Development in Information Retrieval*, (1999) 254-261
8. S. Kullback, J.C. Keegel, and J.H. Kullback. *Topics in Statistical Information Theory*. Springer-Verlag, Berlin Heidelberg New York, (1987)
9. M. Jansen, A. Spink and T. Saracevic. Real Life, real users, and real needs: A study and analysis of user queries on the web. *Information Processing and Management*, 36(2), (2000)
10. C.R. Palmer and J. G. Steffan. Generating Network Topologies That Obey Power Laws. In *Proceedings of Global Internet Symposium*, (2000)
11. The Gnutella protocol specification v0.6. <http://rfcgnutella.sourceforge.net>.

An Algorithmic Approach to High-Level Personalisation of Web Information Systems

Klaus-Dieter Schewe¹ and Bernhard Thalheim²

¹ Massey University, Information Science Research Centre, Private Bag 11 222,
Palmerston North, New Zealand

`k.d.schewe@massey.ac.nz`

² Christian Albrechts University Kiel, Department of Computer Science,
Olshausenstr. 40, D-24098 Kiel, Germany

`thalheim@is.informatik.uni-kiel.de`

Abstract. A web information system (WIS) can be described by abstract locations and transitions between them that are enabled by user actions. This leads to Kleene algebras with tests, thus can be subject to reasoning with equations. In particular, this can be applied to personalisation according to the preferences and goals of users. In this paper we first study decidability and complexity of this reasoning process, then present a Knuth-Bendix-type algorithmic solution for the problem.

1 Introduction

WIS design has to take various aspects such as intention, usage, content, functionality, context and presentation into account [1, 2, 3, 8]. In this paper we will deal with the usage aspect, in particular with the question how a WIS can adapt itself to the preferences and goals of its users, a problem that is commonly referred to as *personalisation*. In continuation of our approach in [8] we start from a system description by a *story space*, which consists of abstract locations called scenes and transitions between these scenes that are enabled by user actions. This can be formalised by an algebraic expression using the theory of Kleene algebras with tests (KATs) [5]. Then user preferences can be modelled by equations, and personalisation becomes a reasoning task with these equations. In this paper we continue this line of research and investigate decidability and complexity of this reasoning process. We will base our study on the results in [6]. The results will have an impact on turning equational reasoning with KATs into an algorithmic approach to personalisation. For this we develop a Knuth-Bendix-type algorithm based on critical-pair completion [4].

2 Storyboarding

The *story space* of a WIS can be described by a edge-labelled directed multi-graph, in which the vertices represent the scenes, and the edges represent transitions between scenes. Looking at this from a different angle, we may concentrate on

the flow of actions. Actions can be treated as being atomic, i.e. we are not yet interested in how an underlying database might be updated. Then each action also belongs to a uniquely determined scene. Actions have pre- and postconditions, so we can use annotations to express conditions that must hold before or after an action is executed. Actions can be executed sequentially or parallel, we must allow choice between actions, and actions can be iterated. As shown in [8] this leads to a *story algebra*. Thus, we can describe a story space by an element of a suitable story algebra, which gives rise to Kleene algebras with tests.

In the following we use $+$ to denote a choice, \cdot to denote a sequence and $*$ to denote iteration. In addition we use 1 for an action that does nothing (usually called skip), and 0 for an undefined action (usually called fail or abort). A *Kleene algebra* (KA) \mathcal{K} consists of a carrier-set K containing at least two different elements 0 and 1 , a unary operation $*$, and two binary operations $+$ and \cdot such that the following axioms are satisfied:

- $+$ and \cdot are associative, i.e. for all $p, q, r \in K$ we must have $p + (q + r) = (p + q) + r$ and $p(qr) = (pq)r$;
- $+$ is commutative and idempotent with 0 as neutral element, i.e. for all $p, q \in K$ we must have $p + q = q + p$, $p + p = p$ and $p + 0 = p$;
- 1 is a neutral element for \cdot , i.e. for all $p \in K$ we must have $p1 = 1p = p$;
- for all $p \in K$ we have $p0 = 0p = 0$;
- \cdot is distributive over $+$, i.e. for all $p, q, r \in K$ we must have $p(q+r) = pq+pr$ and $(p+q)r = pr+qr$;
- p^*q is the least solution x of $q + px \leq x$ and qp^* is the least solution of $q + xp \leq x$, using the partial order $x \leq y \equiv x + y = y$.

A *Kleene algebra with tests* (KAT) \mathcal{K} (see [5]) consists of a Kleene algebra $(K, +, \cdot, *, 0, 1)$, a subset $B \subseteq K$ containing 0 and 1 and closed under $+$ and \cdot , and a unary operation $\bar{}$ on B , such that $(B, +, \cdot, \bar{}, 0, 1)$ forms a Boolean algebra. We write $\mathcal{K} = (K, B, +, \cdot, *, \bar{}, 0, 1)$. Of course the elements of B are the “tests”, and the operator $\bar{}$ expresses negation.

Example 1. Take a simple example, where the WIS is used for ordering products. In this case we have actions $\alpha_1 = \text{select_product}$, $\alpha_2 = \text{payment_by_card}$, $\alpha_3 = \text{payment_by_bank_transfer}$, $\alpha_4 = \text{payment_by_cheque}$, $\alpha_5 = \text{enter_address}$, and $\alpha_6 = \text{confirm_order}$.

Furthermore, use five Boolean conditions: $\varphi_1 = \text{price_in_range}$ to express that the price of the selected product(s) lies within the range of acceptance of credit card payment, $\varphi_2 = \text{payment_by_credit_card}$, $\varphi_3 = \text{payment_by_bank_transfer}$ and $\varphi_4 = \text{payment_by_cheque}$ to express that the user has selected the option to pay by credit card, bank transfer or cheque, respectively, and $\varphi_5 = \text{order_confirmed}$ to express that the user has confirmed the order.

Using these actions and conditions, we can formalise the story space by the algebraic expression $(\alpha_1(\varphi_1\alpha_2\varphi_2 + \alpha_3\varphi_3 + \alpha_4\varphi_4)\alpha_5(\alpha_6\varphi_5 + 1) + 1)^*$.

Using equations we may exploit the axioms of KATs to reason about story spaces. However, we do not just want to derive equations from the KAT axioms,

which would mean to exploit the equational theory of KATs, but we want to derive such equations under the assumption of other equations that describe user preferences and other constraints for the story space expression. In particular, we obtain the following types of equations: A *preference* equation of the form $\varphi(\alpha + \beta) = \varphi\alpha$ expresses that a user conditionally prefers action α over action β , where the condition is expressed by φ . A *precondition* equation of the form $\bar{\varphi}\alpha = 0$ (or equivalently $\alpha = \varphi\alpha$) expresses that the condition φ is a precondition for the action α . A *postcondition* equation of the form $\alpha\bar{\varphi} = 0$ (or equivalently $\alpha = \alpha\varphi$) expresses that the condition φ is a postcondition for the action α . An *invariance* equation of the form $\alpha\varphi = \varphi\alpha$, which is equivalent to $\bar{\varphi}\alpha = \alpha\bar{\varphi}$ and to $\varphi\alpha\bar{\varphi} + \bar{\varphi}\alpha\varphi = 0$, expresses that the condition φ (and so its negation $\bar{\varphi}$) is invariant under the action α . An *exclusion* equation of the form $\varphi\psi = 0$ expresses that the conditions φ and ψ exclude each other. An equation of the form $\alpha\beta = \beta\alpha$ expresses that the order of the actions α and β is irrelevant, hence they behave as if they were executed in parallel.

Then personalisation can be formalised by the following optimisation task: *Given a process $p \in K$ that represents a story space, and a set Σ of equations on K that represents (among other constraints) user preferences and a postcondition $\psi \in B$, we look for a minimal process $p' \in K$ such that $\Sigma \models p\psi = p'\psi$ holds.*

That is, the resulting process p' is a *personalisation* of p according to the *user intention* formalised by ψ .

Example 2. Let us continue Example 1 and localise the story space, i.e. personalise with respect to the location of its users. Say, for a European user the option to pay by cheque should be invalidated. So we obtain the equation $\alpha_4 = 0$. Furthermore, assume that this user prefers to pay by credit card, if this turns out to be a valid option. For this we use the equation $\varphi_1(\alpha_3 + \alpha_4) = 0$, which also implies $\varphi_1\alpha_3 = 0$. Using $\psi = 1$ we can compute $p\psi = (\alpha_1(\varphi_1\alpha_2\varphi_2 + \bar{\varphi}_1\alpha_3\varphi_3)\alpha_5(\alpha_6\varphi_5 + 1) + 1)^*$, i.e. the option to pay by cheque has been removed, and the option to pay by bank transfer has become subject to the condition $\bar{\varphi}_1$, i.e. the price must be out of range for credit card payment.

As in the theory of KATs only completeness and the decidability of decision problems have been investigated so far, we rephrase our problem by the following decision problem: *Given process $p, p' \in K$ such that p represents a story space and $p' \leq p$ holds, and a set Σ of equations on K that represents (among other constraints) user preferences and a postcondition $\psi \in B$, then decide, whether $\Sigma \models p\psi = p'\psi$ holds.*

For the equational theory of KATs we know that it is decidable, but PSPACE-complete. That is, we can decide in polynomial space, whether $p = q$ can be derived from the axioms of KATs. However, the decision problem we are dealing with is of the form $\Sigma \models p = q$ with a set of equations Σ , i.e. in the Horn theory of KATs. From [6] we already know that such problems are undecidable in general. Even if we reduce ourselves to Kleene algebras instead of KATs, we already get Σ_1^0 -completeness, i.e. the problem is in general recursively enumerable hard. However, in our case Σ only contains equations of a particular form, and we

can reduce equations of the form $r = 0$ to the equational theory [7]. Instead of showing $r = 0 \rightarrow p\psi = p'\psi$ we can equivalently show $p\psi + uru = p'\psi + uru$ with $u = (\alpha_1 + \dots + \alpha_m)^*$, where the α_i are all the atomic actions that are not tests.

For conditional preference equations we apply a little trick and introduce exclusive postcondition $\overline{\psi}_\alpha$ and $\overline{\psi}_\beta$ for α and β , respectively. Thus, we have the equations $\psi_\alpha\psi_\beta = 0$, $\overline{\psi}_\alpha\alpha = 0$, and $\overline{\psi}_\beta\beta = 0$, and we can replace the preference equation by $\varphi(\alpha + \beta)\psi_\beta = 0$. For the equations arising from parallelism we apply a different trick replacing $p||p'$ by $pp' + p'p$. These tricks reduce the problem to a decision problem in the equational theory of KATs, which is PSPACE-complete.

3 A Personalisation Algorithm

With respect to the original minimisation problem it seems to be a promising idea to start with the term $p\psi$ and to rewrite the term until we obtain an irreducible term of the form $p'\psi$, i.e. none of the rewrite rules can be applied anymore. If we guarantee that each application of a rewrite rules leads to a smaller term with respect to the order \leq , this irreducible term must be minimal and p' must be the solution to the personalisation problem. The Knuth-Bendix-algorithm from [4] provides the necessary technique and theoretical justification for this approach. Basically, it is a term-rewriting approach that is coupled with so-called “critical-pair completion” (CPC), a method that permits the generation of new rewriting rules during rewriting.

In order to exploit this approach for our purposes here we need a few extensions: For KATs we need an order-sorted approach, as we have a sort B of Booleans, a sort K for the Kleene algebra terms, and $B \leq K$. Then we have to deal with conditional rewrite rules in order to capture the axioms of the star-operator. For this we use co-routines, which validate the preconditions of a rule in a separate rewriting process. Finally, we have to deal with commutativity, in particular with equations arising from parallelism. For this we take the pragmatic approach to use simultaneous rewriting considering all alternatives at the same time.

As standard in order-sorted algebraic specifications we take two *sorts* B and K ordered by $B \leq K$, and the following (nullary, unary, binary) *operators*: $0, 1 : \rightarrow B$, $+, \cdot : K K \rightarrow K$, $*$: $K \rightarrow K$ and $-$: $B \rightarrow B$. Using these sorts and operators we can define *terms* in the usual way. We use p, q, r, \dots (if needed with additional indices) as *variables* of sort K , and a, b, c, \dots (also with indices) as *variables* of sort B . Then each variable of sort B or K is also a term of sort B or K , respectively, 0 and 1 are terms of sort B , $t_1 + t_2$, $t_1 \cdot t_2$, and t^* are terms of sort K , if t , t_1 and t_2 are, \bar{t} is a term of sort B , if t is, and each term of sort B is also a term of sort K .

A *rewrite rule* is an expression of the form $\lambda \rightsquigarrow \varrho$ with terms λ and ϱ of the same sort, such that the variables on the right hand side ϱ are a subset of the variables on the left hand side λ . A *conditional rewrite rule* is an expression of the form $t_1 = t_2 \rightarrow \lambda \rightsquigarrow \varrho$, in which in addition the terms t_1 and t_2 contain the same variables and these form a superset of the set of variables in the left hand side term λ . The application of a rewrite rule $\lambda \rightsquigarrow \varrho$ to a term t is standard: if t

contains a subterm t' that can be matched with λ , i.e. there is a substitution θ such that the application of θ to λ results in t' (denoted $\theta.\lambda = t'$), then replace t' in t by $\theta.\rho$. The application of a conditional rewrite rule $t_1 = t_2 \rightarrow \lambda \rightsquigarrow \rho$ to a term t is defined analogously. In this case we have to show that $\theta.t_1 = \theta.t_2$ can be derived in order to use the substitution θ .

In order to exploit term-rewriting for the personalisation problem we formulate the axioms of KATs and the personalisation equations as (conditional) rewrite rules, then start with $p\psi$ (as in Example 2) and apply the rules until we finally obtain a term of the form $p'\psi$ to which no more rule can be applied. Note that $p\psi$ is closed, i.e. it does not contain variables, so during the whole rewriting process we will only have to deal with closed terms. Following this idea we use the following general (conditional) rewrite rules:

$$\begin{array}{ll}
 p + (q + r) \rightsquigarrow (p + q) + r & p(qr) \rightsquigarrow (pq)r \\
 p + p \rightsquigarrow p & p + 0 \rightsquigarrow p \\
 p1 \rightsquigarrow p & 1p \rightsquigarrow p \\
 p0 \rightsquigarrow 0 & 0p \rightsquigarrow 0 \\
 p(q + r) \rightsquigarrow pq + pr & (p + q)r \rightsquigarrow pr + qr \\
 1 + pp^* + p^* \rightsquigarrow p^* & 1 + p^*p + p^* \rightsquigarrow p^* \\
 pq + q = q \rightarrow p^*q + q \rightsquigarrow q & qp + q = q \rightarrow qp^* + q \rightsquigarrow q \\
 p + q \rightsquigarrow q + p & ab \rightsquigarrow ba \\
 a\bar{a} \rightsquigarrow 0 & \bar{a}a \rightsquigarrow 0 \\
 a + \bar{a} \rightsquigarrow 1 & \bar{a} + a \rightsquigarrow 1
 \end{array}$$

In addition, the personalisation equations from Section 2 give rise to further rewrite rules: conditional preferences $a(p + q) \rightsquigarrow ap$, preconditions $\bar{a}p \rightsquigarrow 0$, postconditions $p\bar{a} = 0$, invariants $ap\bar{a} + \bar{a}pa = 0$, exclusions $ab = 0$, and parallelism $pq \rightsquigarrow qp$. The basic rewriting procedure from [4] uses only unconditional rewrite rules, for which a Church-Rosser property can be shown. For a conditional rewrite rule $t_1 = t_2 \rightarrow \lambda \rightsquigarrow \rho$ whenever λ matches a subterm t' using the substitution θ , the equation $\theta.t_1 = \theta.t_2$ has to be verified. In this case we start a *co-routine*, i.e. a separate rewriting process for $\theta.t_1$ and $\theta.t_2$ to verify this equation. While this co-routine is processed, the rewriting of the original term can be continued until the condition is verified or disproven. In the latter case backtracking has to be applied.

Applying this procedure, the equations in Example 2 represent indeed a rewriting chain, i.e. we can replace all equality symbols by \rightsquigarrow . Note, however that our algorithm would compute much more branches simultaneously, none of which will lead to a different result.

It may happen that the rewriting process gets stuck in the sense that we may rewrite a term t to terms t_1 and t_2 , but neither can we rewrite t_1 to t_2 nor the other way round. In this case t_1 and t_2 represent a *critical pair*, which allows us to add a new rewrite rule $t_1 \rightsquigarrow t_2$ (and its reverse) and continue the rewriting process. This process is called critical-pair completion.

In principle, critical-pair completion can also be applied, when t_1 and t_2 arose from conditional rewriting rules, in which case we would obtain a new conditional rewriting rule. Pragmatically speaking it would be better to wait for the successful termination of the co-routine before adding such conditional rules.

4 Conclusion

In this paper we continued the research from [8] on an inferential approach to personalisation using Kleene algebras with tests (KATs) [5]. Based on the observation that a story space can be represented by a KAT expression, we showed that personalisation leads to an optimisation problem in the Horn theory of KATs. Pragmatically, this can be further translated into a decision problem. While this is undecidable in general, we showed that the special equations we obtain for user preferences and other rules that govern state spaces, we obtain a decidable – yet PSPACE-complete – problem.

In a second step we designed a Knuth-Bendix-type personalisation algorithm that reifies this approach. In addition to the classical approach from [4], which is based on term-rewriting together with critical-pair-completion, we exploited co-routines to cope with conditional rewriting rules. Furthermore, we incorporated the simultaneous handling of several terms in order to capture parallelism. The advantage of our approach to personalisation is that it separates on a high conceptual level the preferences and goals that are the input to personalisation from the story space. As a consequence, personalisation is deduced from these preferences and goal rather than hard-wired into the story space design.

References

- [1] CERI, S., FRATERNALI, P., BONGIO, A., BRAMBILLA, M., COMAI, S., AND MATERA, M. *Designing Data-Intensive Web Applications*. Morgan Kaufmann, San Francisco, 2003.
- [2] DE TROYER, O., AND LEUNE, C. WSDM: A user-centered design method for web sites. In *Computer Networks and ISDN Systems – Proceedings of the 7th International WWW Conference*. Elsevier, 1998, pp. 85–94.
- [3] HOUBEN, G.-J., BARNA, P., FRASINCAR, F., AND VDOVJAK, R. HERA: Development of semantic web information systems. In *Third International Conference on Web Engineering – ICWE 2003* (2003), vol. 2722 of *LNCS*, Springer-Verlag, pp. 529–538.
- [4] KNUTH, D. E., AND BENDIX, P. B. Simple word problems in universal algebras. In *Computational Problems in Abstract Algebra*. Pergamon Press, Oxford, UK, 1970, pp. 263–297.
- [5] KOZEN, D. Kleene algebra with tests. *ACM Transactions on Programming Languages and Systems* 19, 3 (1997), 427–443.
- [6] KOZEN, D. On the complexity of reasoning in Kleene algebra. *Information and Computation* 179, 2 (2002), 152–162.
- [7] KOZEN, D., AND SMITH, F. Kleene algebra with tests: Completeness and decidability. In *Computer Science Logic* (1996), pp. 244–259.
- [8] SCHEWE, K.-D., AND THALHEIM, B. Conceptual modelling of web information systems. *Data and Knowledge Engineering* 54, 2 (2005), 147–188.

sPAC (Web Services Performance Analysis Center): A Performance-Aware Web Service Composition Tool

Hyung Gi Song¹ and Kangsun Lee^{2,*}

¹ R&D Institute, Netville Co., Ltd., 161-7 Yeomni Mapo, Seoul,
121874 South Korea

² Dept. of Computer Engineering, Myongji University, San 38-2 Namdong Yongin,
Kyungki, 449728 South Korea
Tel: +82-31-330-6444; Fax: +82-31-330-6432
ksl@mju.ac.kr

Abstract. Web services and their composition (web processes) are promising technologies to efficiently integrate disparate software components over various types of systems. As many web services are nowadays available on Internet, quality of services (QoS) and performance/cost become increasingly important to differentiating between similar service providers. In this work, we introduce sPAC (Web Services Performance Analysis Centre) and show how customers can benefit from sPAC to consider performance in composing and commercializing web services.

Keywords: Web Services and information management.

1 Introduction

As many web services with similar functionalities are available on the Internet, quality of services (QoS), for example, availability, security, reliability and performance, will distinguish service providers from each other when a customer selects suitable web services during web services composition [1,2]. QoS properties are inherently dynamic and change in real time depending on how these services are actually performing. Among the dynamic QoS factors, performance has been the most difficult factor to access, since it involves non-deterministic networks, abrupt changes on load intensity, and unexpected usage patterns [3-5].

In this work, we present sPAC (web Services Performance Analysis Center), a performance-aware web services composition environment. sPAC helps users to graphically design a new web process, analyze performance properties of the web process and reengineer the web process, accordingly. sPAC analyzes the performance of the web process cost-effectively by mixing test and simulation-based method; Test-based analysis is performed in low load conditions, while simulation-based method is carried out for heavy load conditions. Increased accuracy can be achieved by carrying out simulations based on test results; Test-based analysis records how the web process

* Corresponding author.

behaves on low load conditions in terms of response time and throughput, and these historical data are used to set up simulation parameters.

This paper is organized as follows. In Section 2, sPAC methodology and performance metrics are introduced with detailed explanation on key components. Section 3 demonstrates sPAC with an example and shows how users can benefit from sPAC to design, reengineer and verify their new web process. Section 4 concludes this paper with future works to achieve.

2 sPAC (Web Services Performance Analysis Centre)

sPAC works with the perspective of service consumers and mixes simulation and test-based analysis method to evaluate and estimate the performance of web services. In this section, we present performance metrics and the methodology of sPAC (Web Services Performance Analysis Centre).

2.1 sPAC Performance Metrics

DRT (Dissected Response Time) and TRT (Traced Response Time) are the performance metrics of sPAC.

DRT divides response time into three factors: *Network Time* (N), *Messaging Time* (M) and *Service Time* (S). The response time, T, for a single web service, s, is defined in Equation 1.

$$T(s) = N(s) + M(s) + S(s) \quad (1)$$

Network Time is the amount of delay determined by bandwidth of network path between customers and the providers of web services, network traffic and performance of network equipments. *Messaging Time* is the amount of time taken by service providers to process SOAP messages. *Service Time* is the amount of time for a web service to perform its designated task. It depends on efficiency of business logic, hardware capability, framework for web services and/or operating system of web services.

When a web process is commercialized with packaged software or in the form of web application, each web service is expected to experience heavy load intensity. TRT performance analysis creates *virtual* users with Java threads, lets them invoke the web services simultaneously with the specified order and manner, and collects DRT for various load conditions.

2.2 sPAC Methodology

As shown in Figure 1, our methodology conducts performance analysis in dual mode in order to save time and cost: test-based analysis for low load intensity and simulation-based analysis for heavy load intensity. Web services are automatically translated into a discrete event simulation model, while the results from test-based analysis are fed into the simulation parameters to increase estimation accuracy.

A customer defines how web services are formed into a new web process using UML's activity diagram [6]. With activity diagrams, web services are represented as

nodes, while the flow between them is represented as links with decorations to specify fork, join, execution types (parallel or serial), and other various conditions. Then, Web services are dynamically invoked and executed for test-based performance analysis. DRT (Dissected Response Time) and TRT (Traced Response Time) are analyzed under low load intensity, and recorded in history database. Meanwhile, the web process is automatically translated into a simulation model. Our simulation model is constructed based on Simjava, a process-based discrete event simulation package for Java [7]. A Simjava simulation is a collection of entities each of which runs in its own thread. These entities are connected together by ports and can communicate with each other by sending and receiving event objects through these ports. During the translation phase of the activity diagram, nodes and links of the activity diagram are represented as entities and ports, respectively, in SimJava. The entities' ports are linked together by following the execution order in the activity diagram. For better accuracy, results from test-based performance analysis are retrieved from the history database, and normalized for the average response time to process a single user request during the test analysis. Then, the resulting average response time and variance are utilized as the delay amount of time to send and receive event objects during the simulation analysis. Finally, DRT, TRT and TPM (Transactions per Minute) results are reported with text and graphical forms. These results help users determine if the web process satisfies the performance criteria.

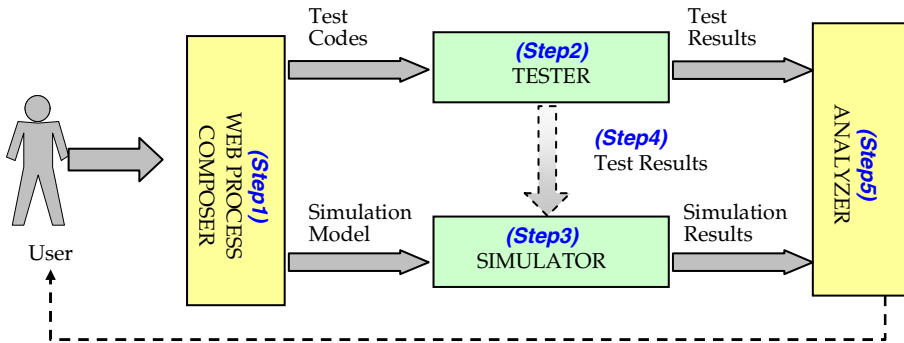


Fig. 1. Performance Analysis Methodology

3 Example: My Travel Planner

In this section, we illustrate sPAC with an example of *My Travel Planner*. Suppose *My Travel Planner* will provide services of booking a flight, reserving an accommodation, renting a car, finding out the current currency rate between Korea and China, exchanging travel money, and processing credit cards. Also, we would like to create *My Travel Planner* just by integrating the existing web services available on the web. As shown in Figure 2, users can search available web services in UDDI and graphically specify how the web services are formed into a new web process.

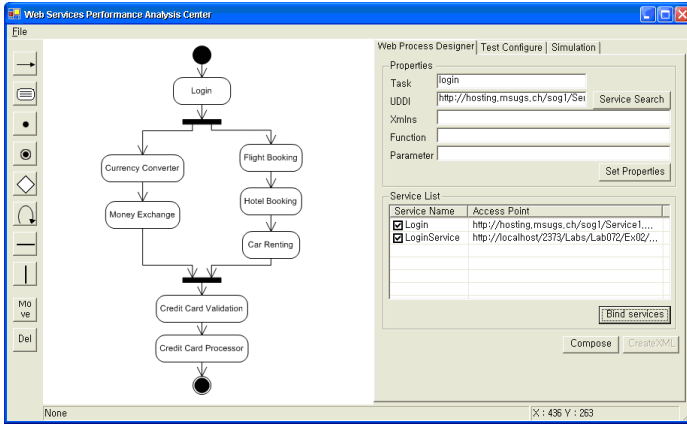
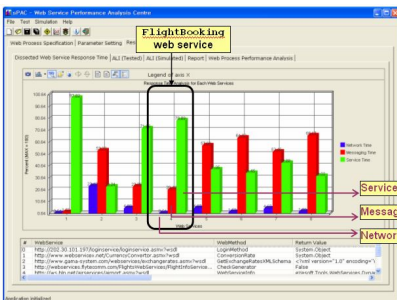


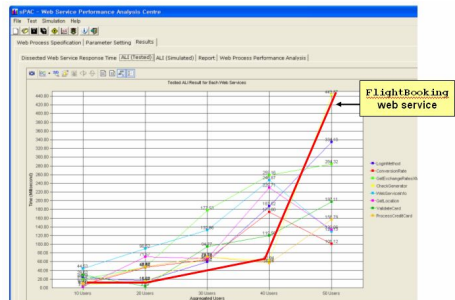
Fig. 2. Web Process Specification

Based on the flow between the selected web services, the input and output relationships are checked to confirm if output values of a web service can be used as input values for the successor service. Then, execution parameters (for example, input data) and experimental parameters (for example, minimum and maximum number of simultaneous request, and testing and simulation duration) are set by users.

Then, sPAC conducts DRT and TRT tests for the given web process with small number of simultaneous requests. In *My Travel Planner* example, DRT and TRT tests are conducted with 10 – 50 numbers of simultaneous users, and take 40 second to complete. Figure 3 shows DRT and TRT test results, respectively, for *My Travel Planner* example. For example, FlightBooking web service becomes the performance bottleneck of *My Travel Planner* as the number of simultaneous requests increases as shown in Figure 3. According to DRT tests in Figure 3, the response time of FlightBooking web service is mainly dominated by service time (78.89% of the response time) compared to messaging time (20.46 %) and network time (0.64%). This observation suggests us to reengineer the business logic of FlightBooking, or to increase hardware capability, or to find other alternatives.



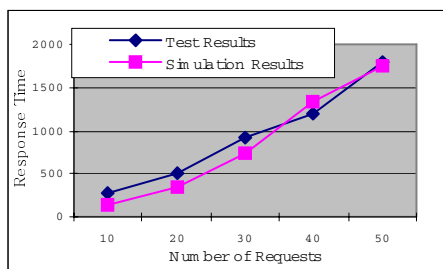
(a) DRT test results



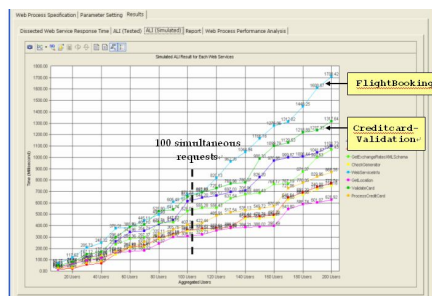
(b) TRT test results

Fig. 3. Test-based performance analysis of My Travel Planner

Meanwhile, sPAC translates the activity diagram into a SimJava-based simulation model. Also, DRT and TRT test results of each web service are analyzed for the average response time to process a single request. The average values are utilized in a simulation model as the amount of delay to send and receive event objects. Figure 4 shows the difference between test-based analysis and simulation-based analysis under the low load intensity of 10 – 50 simultaneous requests. The graph indicates that our simulation-based analysis is accurate enough to further estimate the performance of *My Travel Planner* under the heavy load intensity of 50 – 200 simultaneous requests.



(a) Test results vs. Simulation Results



(b) TRT estimation

Fig. 4. Simulation-based performance estimation of *My Travel Planner*

Figure 4 shows simulation-based TRT estimation for the web services. Under heavy load of 200 simultaneous requests, FlightBooking and Creditcard-Validation are expected to be the performance bottlenecks. Also, Figure 4 indicates that the overall response time of *My Travel Planner* drastically increases as the number of simultaneous requests exceeds 100. All the estimation data will be used for software architects to foresee the performance of *My Travel Planner* after deployment, and to decide hardware configuration, accordingly.

4 Conclusion

In this paper, we introduced sPAC, a performance-aware web service composition tool. In order to save time and cost, performance analysis has been done in dual mode: test-based mode for low load intensity and simulation-based mode for heavy load intensity. The given web process was automatically translated into a process-based discrete event simulation model, while the results from test-based analysis were fed into the simulation parameters to increase estimation accuracy. We will extend our QoS analysis methods to other dynamic QoS properties, such as, availability, reliability, security and their combinations for future work.

Acknowledgement

This work was supported by grant No. R05-2004-000-11329-0 from Korea Research Foundation.

References

1. Eric Newcomer, *Understanding Web Services: XML, WSDL, SOAP and, UDDI*, Addison-Wesley, 2002
2. W3C Working Group, *QoS for Web Services: Requirements and Possible Approaches*, <http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/>, 2003
3. Daniel A. Menasce and Virgilio A.F. Almeida, *Capacity Planning for Web Services: Metrics, Models, and Methods*, Prentice-Hall, 2002
4. J.D. Meise, Srinaath Vasireddy, Ashish Babbar, Alex Mackman, *How to: Use ACT to Test Web Services Performance*, Microsoft Developer Network, Microsoft Corporation, 2004
5. Gregory Silver, John A. Miller, Jorge Gardoso, Amit P. Sheth, Web service technologies and their synergy with simulation, In Proceedings of the 2002 Winter Simulation Conference, pp. 606 – 615
6. Object Management Group, *UML (Unified Modeling Language) TM Resource Page*, <http://www.uml.org/>, January 2005
7. Fred Howell, Ross McNab, *Simjava Library*, <http://www.dcs.ed.ac.uk/home/hase/simjava>, 1996

Design and Implementation of a Service Bundle Manager for Home Network Middleware

Minwoo Son¹, Jonghwa Choi¹, Namhoon Kim², Dongkyoo Shin¹, and Dongil Shin^{1,*}

¹Department of Computer Science and Engineering, Sejong University
98 Kunja-Dong, Kwangjin-Ku, Seoul 143-747, Korea
{minwoo15, com97, shindk, dshin}@gce.sejong.ac.kr

²School of Computing and Information, Dongyang Technical College
62-160 Kochuk-Dong, Kuro-Ku, Seoul 152-714, Korea
nhkim@dongyang.ac.kr

Abstract. Since different kinds of home network technologies coexist, seamless connections between different kinds of network protocols are essential for home computing environments. Several kinds of home network middleware have been developed and OSGi (Open Service Gateway Initiative) was initiated mainly for interoperability support with service distribution between various middleware environments. In this paper, we analyze the weaknesses in the OSGi service platform, such as a non-distributed framework, passive user management, device management, and service bundle management. Moreover, we propose SBM (Service Bundle Manager) to efficiently manage various service bundles. This paper concludes with the implementation results for SBM.

1 Introduction

Ubiquitous Computing [1] means that users can use computers naturally and conveniently in life, regardless of place and time. It means that a computer existing anywhere can use specialized services, changing its contents according to place or time via sensing and tracking. A representative example of Ubiquitous Computing is a Home network [2], which refers to a computing environment based on information from home appliances, in which a user can connect to the internet through a wired/wireless network.

Home network technologies include Home Gateway [3], Home Server [4] and Middleware that controls home appliances, facilitates interaction between electronics, and supports various services. A variety of middleware for home networks have been developed, including UPnP(Universal Plug and Play) [5,6], Jini [7,8], HAVi(Home Audio Video Interoperability) [9,10], IEEE 1394 [11,12], Power Line Communication[13], and so on. The existing home network middleware has problems such as lack of interoperability and difficulty of distributing new middleware-specific services. OSGi (Open Service Gateway Initiative) [14,15] has been developed to overcome these problems by enabling the easy deployment of services for local home network.

* Correspondence author.

OSGi Spec. version 3 includes many services. For example it includes Framework for a service bundle manager and event processing, Log Service for logging information and retrieving current or previously recorded log information in the OSGi Service Platform, Device Access Service for an electronic home appliance manager and for recognizing new home appliances, supporting appropriate device drivers, and so on.

But the OSGi Service Platform doesn't support updating, installing, removal, etc. for the active life-cycle of service bundles, and will not automatically check-in a device's state, update a device driver, distributed framework, and so on. Therefore we suggest SBM (Service Bundle Manager) to solve these problems.

This paper is composed of five sections. Section 2 introduces OSGi. In Section 3, we propose SBM to efficiently manage many kinds of service bundles based on OSGi and describe related implementation results in Section 4. Finally we conclude in Section 5.

2 Background

OSGi was created in 1999 in order to define an open standard for the delivery of services in networked environments, e.g. vehicles, homes etc. and was supposed to solve problems involving the interaction between several kinds of home network middleware and service distribution etc. The OSGi service platform is divided into two parts, the OSGi Service Framework and OSGi Service. The OSGi framework supports service registry, life-cycle management of service bundles etc. and includes registry service, persistent, data storage and life-cycle management for a service, etc. in an extensible Java runtime environment.

The OSGi framework supports an execution environment for services. An OSGi service is defined by a Java Interface. OSGi services include HTTP, Logging, Device Access Service, etc. A service is implemented as a bundle. A bundle is the smallest unit of management for the Framework and the framework manages its installation, uninstall, resolving, stopping, starting, and active life-cycle. A bundle consists of java class code files and additional resources, etc. In addition, the framework has Manifest file Meta information for the bundle or a bundle's install, start, stop etc., information. Several OSGi services may be include on a bundle, and form a standard unit of distribution and management.

3 SBM (Service Bundle Manager) Design

Figure 1 shows SBM architecture. To control home appliances, user uses two connection-systems. The first method makes it possible for a user to control a service bundle in SBM, after the user is authenticated through a web browser which uses web service. The second method is a UIML (User Interface Markup Language) [20] document that transmits using Mobile Devices such as PDA or Web PAD using a Network Service to Service Bundle Manager Server approach. A UIML document is stored to Service Using History Storage and the document is analyzed. The UIML document pattern is made according to the data form of the UIML document to divide electronic devices into image devices and sound devices.

To control electronic devices through using a Web or Mobile Device, it accepts data on access privileges by a user's Device in User Manager through a user ID if users approach. After Device Manager receives an electronic device ID and device function services, it finds an appropriate driver through the device. Finally, users can control devices by service bundles.

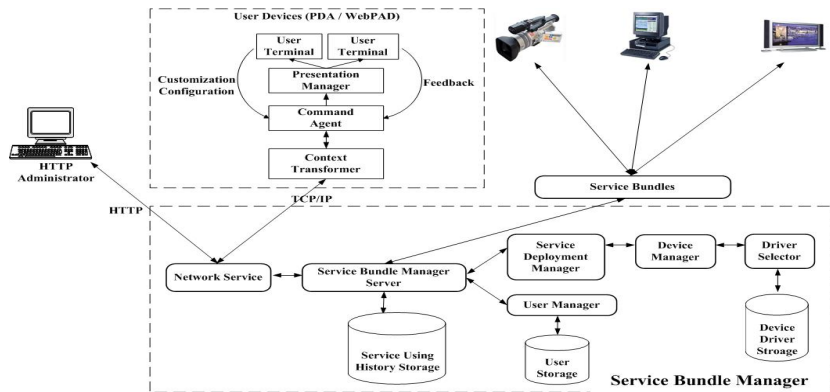


Fig. 1. Service Bundle Manager Architecture

Here introduces briefly and to services of Manager and Storage part in SBM. First we introduce Manager Part. Network Service offers user to connect SBM through Web, TCP/IP and to support service that controls service bundle. Service Bundle Manager Server manages execution life-cycle of service bundle (install, start stop, resume and uninstall) and collects service bundle's information. Also it sends state information of service bundle to Administrator on schedule and is managed service bundle through network by user. Service Deployment Manager links driver with fitting service bundle. Device Manager manages each device's driver and service (addition, insert and remove) and periodically updates driver and stores driver in driver storage. Driver Selector selects best suited to a driver service which user wants. User Manager manages user's personal information (addition, insert and remove). Next we introduce Storage Part. Service Using History stores UIML, which mobile device sends, to control device etc. User Storage stores users' personal information such as id, name, age, career, and so on. Device Driver Storage save driver of each device.

4 Implementation of Service Bundles Based on SBM

This paper utilized an OSGi Release 3 compliant environment.

4.1 Web Application Service Bundle

Figure 2 shows the Web Application Service Bundle hierarchy. Web Application obtains the application's function transmission and connecting components. An appli-

cation user uses a web browser for using a web application’s function. Users view contents, according to fill data values and click links, through a browser.

The WebServer class offers multimedia content services such as images, videos or audio files and provides HTML Tag page services through HtmlCon. Application-Server class services HTML (HyperText Markup Language) Tag through WebServer class. CodeCon class manages web services such as XML (eXtensible Markup Language), ASP (Active server Pages) and PHP.

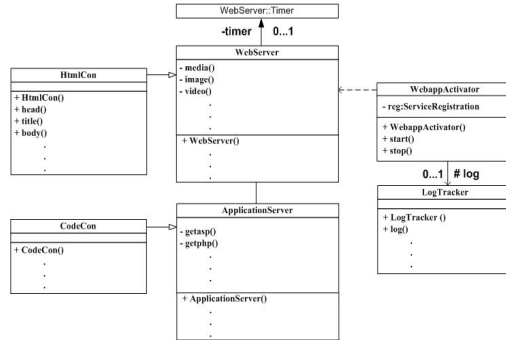


Fig. 2. Web Application Service Bundle hierarchy

4.2 Camera Control Service Bundle

Camera Control Service views a home’s state in real-time through the web, and provides control functions such as moving a Camera’s lens, lens Zoom-in/out, etc.

The Activator component and Event Broker component approach the SBM framework as shown Figure 3. An Activator component, by interaction with the SBM framework, provides know-how to do the Camera Control Service Bundle’s start and stop from the administration interface of the framework. The Related Client in Camera Control Service Bundle Event records in Event Broker.

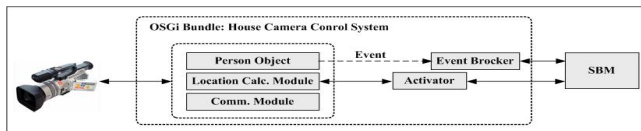


Fig. 3. Camera Control Service Bundle Architecture

Figure 4 shows each class’s relation in Camera Control Service Bundle. CamActivator class controls a bundle’s states, such as start, install and stop, etc. If calling start(), a service bundle scarcely starts when CamFrame class performs. After calling the connect() method, the CamFrame class controls an image’s information and mode. The useZoom() method sets up zoom lens. Methods for checking panning, tilting and zooming capabilities of the camera are also implemented.

LogTracker Class processes the recording of events and errors. The log() method logs a message with an exception associated with a specific service. The class starts as

soon as the service bundle starts and if the service bundle stops, the class calls the `log.close()` method to stop. Figure 5 shows the transmission of data between Camera Control Service and Client through the Client class to control the Camera's Channel during the Camera Control Service Bundle's run-time.

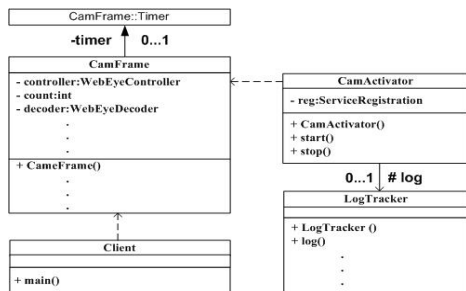


Fig. 4. Camera Control Service Bundle hierarchy

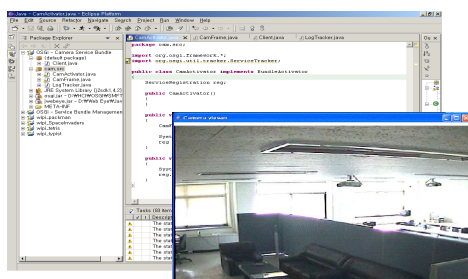


Fig. 5. View Camera Control Service Bundle

5 Conclusion

This paper proposes SBM, which efficiently manages several service bundles based on OSGi and describes the execution conclusion of service bundles in an SBM environment. We designed SBM in order to efficiently manage many service bundles, based on the OSGi service platform.

SBM, which solves the OSGi service platform's weaknesses, such as User Management and Device Management, permits certified users to control each device and automatically designs a service for each device. After a user enters SBM using a web service and mobile device for the control of a device, SBM controls the sending of the device's service information, which analyses access privileges through User Manager and Device Manager, to the server. We did research on service bundles in a home network system and on a manager for home appliances' control and web application service. SBM updates service bundles automatically and efficiently manages service bundles through managing a user's authorization and each device's control.

Future work will be done on a study of SBM to efficiently manage home appliances and service bundles, after extending the services of SBM such as Context Awareness, authenticated security and distribution.

References

1. Schulzrinne. H., Xiaotao. Wu, Sidiroglou. S., Berger. S.: Ubiquitous computing in home networks, *Communications Magazine, IEEE*, Vol. 41, Issue. 11, (2003) 128 - 135
2. Rose B.: Home networks: a standards perspective, *Communications Magazine, IEEE*, Vol. 39, Issue. 12, (2001) 78 - 85
3. Saito. T., Tomoda. I., Tokabatake. Y., Arni. J., Teramoto. K.: Home gateway architecture and its implementation, *Consumer Electronics 2003, ICCE 2003, IEEE International Conference on*, (2003) 386-387
4. Changseok Bea, Jinho Yoo, Kyuchang Kang, Yoonsik Choe, Jeunwoo Lee: Home server for home digital service environments, *Consumer Electronics, IEEE Transactions on*, Vol. 49, Issue. 4, (2003) 1129-1135
5. UPnP Specification v1.0 homepage, <http://www.upnp.org>
6. Dong-Sung Kim, Jae-Min Lee, Wook Hyun Kwon, In Kwan Yuh: Design and implementation of home network systems using UPnP middleware for networked appliances, *Consumer Electronics, IEEE Transactions on*, Vol. 48, Issue. 4, (2002) 963 - 972
7. Jini Specification v1.0 homepage, <http://www.jini.org>
8. Landis. S., Vasudevan. V.: Reaching out to the cell phone with Jini, *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, (2002), 3821-3830
9. HAVi Specification v1.1 homepage, <http://www.havi.org>
10. Lea. R., Gibbs. S., Dara-Abrams. A., Eytchison. E.: Networking home entertainment devices with HAVi, *Computer*, Vol. 33, Issue. 9, (2000) 35-43
11. IEEE 1394 Specification v1.0 homepage, <http://www.1394ta.org>
12. Nakagawa. M., Honggang Zhang, Sato. H.: Ubiquitous homelinks based on IEEE 1394 and ultra wideband solutions, *Communications Magazine, IEEE*, Vol. 41, Issue. 4, (2003) 74 - 82
13. Ferreira. H.C., Grove. H.M., Hooijen. O., Han Vinck, A.J.: Power line communications: an overview, *AFRICON, IEEE AFRICON 4th*, Vol. 2, (1996) 558 - 563
14. OSGi Specification v. 3.0, March.2003 homepage, <http://www.osgi.org>
15. Young-Gab Kim, Chang-Joo Moon, Dae-Ha Park, Doo-Kwon Baik: A Service Bundle Authentication Mechanism in the OSGi Service Platform, *Advanced Information Networking and Applications, 2004. AINA 2004. 18th International Conference on*, Vol. 1, (2004) 29-31
16. Radio Frequency Identification (RFID) homepage, <http://www.aimglobal.org/technologies/rfid>
17. Want. R.: Enabling ubiquitous sensing with RFID, *Computer*, Vol. 37, Issue. 4, (2004) 84-86
18. Bluetooth homepage, <http://www.bluetooth.com>
19. Kwang Yeol Lee, Jea Weon Choi: Remote-controlled home automation system via Bluetooth home network, *SICE 2003 Annual Conference*, Vol. 3, (2003) 2824-2829
20. UIML homepage, <http://www.uiml.org>

A State-Transfer-Based Dynamic Policy Approach for Constraints in RBAC

Cheng Zang, Zhongdong Huang, Gang Chen, and Jinxiang Dong

College of Computer Science, Zhejiang Univ.,
Hangzhou, China.P.R. 310027
zang623@tom.com
{hzd, cg, djx}@cs.zju.edu.cn

Abstract. RBAC is widely used in access control field, and this paper proposes an approach to implement dynamic policy transfer on this model. Our approach monitors state-transfers of subjects and transfers policies correspondingly. It holds a finite number of states and a policy transfer set containing the predefined policies. When a state-transfer occurs, an appropriate policy chosen from the policy transfer set will be applied to change the user-role mapping or the role-permission mapping from one to another. This policy transfer not only focuses on the current state, but also takes the previous state into consideration since changing from different state will lead to a different current policy.

1 Introduction

The role based access control (RBAC) model[3] was proposed by Sandhu et al.. This model maps users to roles and roles to permissions, making policy authorization simplified since the amount of roles are much fewer than users. Also, the model makes roles inheritable to enhance the efficiency of policy management.

Based on RBAC, methods are widely developed to extend this model and make it more flexible, for example a dynamic feature. TRBAC [1] and GTRBAC [2] are developed to enhance RBAC model with a temporal user-role mapping extension which allows users to assign to different roles at different time.

In [4] and [5] the policy is context-based or history-based, but context or history information in these approaches mainly focuses on the user's behavior or has to be recorded in a log file or database.

But in some situation not only the user/role's historical behavior can have influence on the current state, but also the environment value, the previous policy and so on can affect the current state too. What's more, even in a same state, the policy could be different due to the different previous state. Consider an admiral and a rear-admiral, and suppose the rear-admiral has been promoted to vice-admiral, while the admiral has been picked a star from his shoulder. At this time they are both vice-admirals, so they have the same authorization to get access to the secret files, to command the fleets, to earn the same salary. But the former one could be promoted to admiral further while the latter one will probably not be promoted for ever. We are going to develop an easy way to describe the influence from the previous state including all the changeable elements without the log or database recording the historical information.

In this paper, we propose a state-transfer-based dynamic policy approach for constraints in RBAC. It focuses on the state-transfer including the resource load changing, the time changing, the user-role mapping changing, the role-permission mapping changing, etc. Generally speaking, it covers almost all the elements in an RBAC model.

The remainder of this paper is organized as follows: In Section 2, the approach is detailed and some features are explained. Section 3 gives an example to show how this approach monitors the state-transfer and changes the policies dynamically. Related works are discussed in Section 4.

2 The Approach

2.1 State, State-Transfer and Policy

State-transfer is the base of this approach, so it's necessary to define the state and state-transfer explicitly.

There are users, roles, objects, permissions and the mappings between them in RBAC model. To define a state explicitly, we need to include all the variables that can uniquely decide a certain state into the state description, also we need to take the environment into consideration too. To unify the definition of all these elements, we consider all users, roles, objects, permissions and environment as *Subjects*. That is when talking about the role-state or the object-state, we will use the subject-state instead.

Definition 1. *Attribute is a set containing natural characteristics of a subject.*

Attributes contain all the elements by which we can describe a subject, such as the IP address, the power ON/OFF state, the CPU load, the environment time, etc.

Definition 2. *Partial State is a set containing the values of attributes of a subject.*

In this approach, a state should contain the values of attributes and relationships between subjects. So, we define a set contains only values of attributes as partial state.

Definition 3. *Partial-State-Based Policy (PSBpolicy) is a set containing mapping rules which is certain according to a certain partial state.*

PSBpolicy contains mapping rules like user-role or role-permission mappings. Policies defined as PSBpolicy will be certain when state is decided. That is, the PSBpolicy is independent of the previous state from which the current state transfers.

Definition 4. *State is a set composed of partial state and PSBpolicy.*

A complete definition of a state in this approach contains all elements that can describe a certain state.

Definition 5. *State-Transfer is a process that one state changes to another one.*

When some values of attributes or PSBpolicy changes, we say a state-transfer occurs. For example, the time changes, the role assignment changes, etc.

Definition 6. *State-Transfer-Based Policy (STBpolicy) is a set containing mapping rules which is decided by a pair of previous state and current state, and it is uncertain corresponding to a single certain state.*

STBpolicy also contains the mapping rules, but policies defined as STBpolicy will be uncertain in a certain state, since it also depends on the previous state.

Definition 7. Policy is a set containing PSBpolicy and STBpolicy which can be changed according to a state-transfer.

STBpolicy is changed according to the pair of previous state and current state, and PSBpolicy is changed according to the current partial state. When a state-transfer occurs, both of them can be triggered, we call it a policy-transfer.

Here we draw a graph in Fig 1 to illustrate the relationship between the elements defined above.

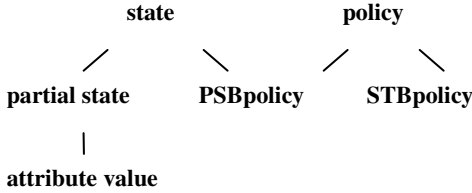


Fig. 1. The containing relationship

2.2 Expressions

In this section we will define a cluster of operations to explain how an appropriate policy is chosen from policy transfer set.

Table 1 defines a list of state predicates in our approach. SBJ is a subject set, PST is a partial state set, ST is a state set, and P is a policy set (both PSBpolicy and STBpolicy belong to the policy set). $sbj \in SBJ$, $pst \in PST$, $st_p, st_c \in ST$, $p \in P$. And, two special states in ST is defined as $*_0 \in ST$, $*$ represents any state in ST, and 0 represents null state. For example, $*$ can represents $st_1, st_2 \dots st_n$, and 0 stands for a null state like before initiation or after end. We can see that the pair $(st_p, *)$ means state st_p changes to any other state while $(*, st_c)$ means a state-transfer from any state to st_c . and they represent state breaking and state obtaining.

Table 1. Various state predicates

predicate	evaluation domain	semantic
$previous_state_decided(sbj, st_p)$	$SBJ \times ST$	st_p is the previous state of sbj
$current_state_decided(sbj, st_c)$	$SBJ \times ST$	st_c is the current state of sbj
$state_transfer_decided(sbj, st_p, st_c)$	$SBJ \times ST \times ST$	state of sbj changes from st_p to st_c
$STBpolicy_defined(sbj, st_p, st_c, p)$	$SBJ \times ST \times ST \times P$	STBpolicy defined for state transfer from st_p to st_c
$policy_decided(sbj, p)$	$SBJ \times P$	policy decided for sbj
$partial_state_decided(sbj, pst)$	$SBJ \times PST$	pst is the partial state of sbj
$PSBpolicy_defined(sbj, pst, p)$	$SBJ \times PST \times P$	PSBpolicy defined for partial state pst .
$PSBpolicy_decided(sbj, p)$	$SBJ \times P$	PSBpolicy decided for sbj

Definition 8.

1. $\text{partial_state_decided}(\text{sbj}, \text{pst}) \wedge \text{PSBpolicy_defined}(\text{sbj}, \text{pst}, \text{p})$
 $\rightarrow \text{PSBpolicy_decided}(\text{sbj}, \text{p})$
2. $\text{PSBpolicy_decided}(\text{sbj}, \text{p}) \wedge \text{partial_state_decided}(\text{sbj}, \text{pst})$
 $\rightarrow \text{current_state_decided}(\text{sbj}, \text{stc})$
3. $\text{current_state_decided}(\text{sbj}, \text{stc}) \wedge \text{previous_state_decided}(\text{sbj}, \text{stp})$
 $\rightarrow \text{state_transfer_decided}(\text{sbj}, \text{stp}, \text{stc})$
4. $\text{state_transfer_decided}(\text{sbj}, \text{stp}, \text{stc}) \wedge \text{STBpolicy_defined}(\text{sbj}, \text{stp}, \text{stc}, \text{p})$
 $\rightarrow \text{policy_decided}(\text{sbj}, \text{p})$

In Definition 8, statement 1 means a PSBpolicy can be decided if a partial state is decided and a PSBpolicy according to this partial state is predefined. Statement 2 implies that a state is decided by both partial state and PSBpolicy. Statement 3 shows that a state-transfer is decided when both a previous and a current state are decided. And statement 4 indicates that a policy is decided if a state-transfer and a STBpolicy according to this state-transfer is predefined.

3 An Example

We assume that there is an information service in the LAN, and using this resource needs two steps: logon and access the information. The CPU load of this service is appropriate between 20%-50%, since high CPU load may cause other services jammed and low usage may waste the capacity of CPU. Also, there are two groups of users in LAN who may use the resource. One is high_priority_user (HPU) and the other one is low_priority_user (LPU). Each user belongs to one and only one of them. When the CPU load is over 50%, both of them can logon but only HPU can get access to the information, and when the CPU load is lower than 20%, both of them are authorized to logon and access the information. Now the problem remaining is what is the policy when CPU load is between 20% and 50%? Usually people tend to do the same thing together, for example, at daytime the office LAN is always busy but at night it is idle since employees work at the same time and rest at the same time, or when a big news is published, people all go to visit web pages, and some days later they take no interest any more. So here we consider that when the usage of this resource is increasing, it tends to keep increasing since it may indicate that the more access is coming, on the other hand, if the usage is decreasing, it tends to keep decreasing too. Thus we define the policy in Table 2.

We have mentioned policies in state A and C before, so here we only explain the policy in state B. When the CPU load is between 20% and 50%, the PSBpolicy is decided as “both of them can logon, and HPU can access the information.” The STBpolicy is decided by both previous and current state, describing as “LPU can access the information according to a state transfer from C to B” and “LPU cannot access the information according to a state transfer from A to B”. As we assumed above, When CPU load increasing from <20% to 20%-50%, it tends to increase to >50%, so the STBpolicy “LPU can not access the information” tries to prevent the load from keeping increasing. Fig.2a shows the result due to the STBpolicy. We can see the tendency line is above the curve using STBpolicy because the policy denied the access from LPU and

only the HPU keeps increasing. Also, Fig.2b shows the contrast situation. These STBpolicies obviously maintain the CPU load in the appropriate usage for a longer time.

Table 2. Policy defined

state A	partial state	CPU load < 20%
	PSBpolicy	HPU logon = true; LPU logon = true HPU access= true; LPU access= true
	STBpolicy in state A	null
state B	partial state	CPU load 20%-50%
	PSBpolicy	HPU logon = true; LPU logon = true HPU access= true
	STBpolicy in state B	state transfers from C to B: LPU access= true
		state transfers from A to B: LPU access= false
state C	partial state	CPU load > 50%
	PSBpolicy	HPU logon = true; LPU logon = true HPU access= true; LPU access= false
	STBpolicy in state C	null

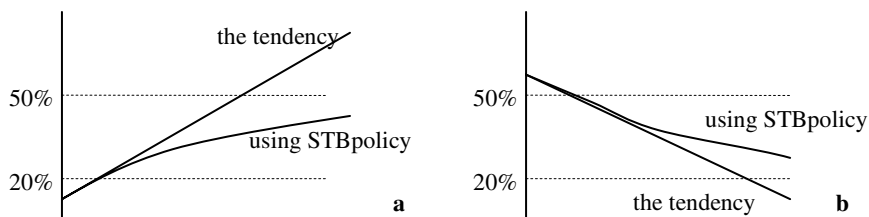


Fig. 2. The diagram of the example

4 Related Works

Dynamic policy makes access control flexible and extensible, and also makes it easier to adapt to various situations. Various solutions are proposed to achieve this purpose. In [6], an entire GTRBAC model is proposed based on many former researches on temporal policy, which makes policies dynamic according to different time.

Some other dynamic solutions are based on all kinds of elements besides time. In [4][7][9], they propose dynamic solutions that decide a policy according to different situations such as different time, IP, role, action, etc. these make access control policy be able to use appropriate rules to dominate the system at different situations. In paper [8] a context-based identification mechanism is hired to implement a dynamic separation of duty built on a web security purpose. Users' historical actions are used to figure out the potential risk in [5], in this way, the policies may change based on the log of actions.

References

1. Bertino E, Bonatti P A, Ferrari E. TRBAC:A temporal role based access control model[J]. ACM Transactions on Information and System Security, August 2001 ,4(3):191~233.
2. Joshi J B D , Bertino E, Ghafoor A.Temporal hierarchies and inheritance semantics for GTRBAC[C].Seventh ACM symposium on Access Control Model and Technologies, June 2002,74~83.
3. SANDHU R S, COYNE E J, FEINSTEIN H L, et al. Role-based access control models[J]. IEEE Computer, 1996, 29(2):38-47.
4. Gustaf Neumann, Mark Strembeck. An Approach to Engineer and Enforce Context Constraints in an RBAC Environment. SACMAT'03, June 2-3, 2003, Pages: 65 – 79.
5. G. Edjlali, A. Acharya, and V. Chaudhary. History-based Access Control for Mobile Code. InProc. of the Fifth ACM Conference on Computer and Communications Security (CCS), November 1998.
6. Joshi, J.B.D.; Bertino, E.; Latif, U.; Ghafoor, A.; A generalized temporal role-based access control model.Knowledge and Data Engineering, IEEE Transactions on Volume 17, Issue 1, Jan. 2005 Page(s):4 - 23
7. Yamazaki, W.; Hiraishi, H.; Mizoguchi, F.; Designing an Agent-Based RBAC System for Dynamic Security Policy Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. 13th IEEE International Workshops on 14-16 June 2004 Page(s):199 - 204
8. Wolf, R.; Keinz, T.; Schneider, M.; A model for content-dependent access control for Web-based services with role-based approach .Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on 1-5 Sept. 2003 Page(s):209 - 214
9. Bhatti, R.; Bertino, E.; Ghafoor, A.; A trust-based context-aware access control model for Web-services. Web Services, 2004. Proceedings. IEEE International Conference on 6-9 July 2004 Page(s):184 - 191

A New Cache Model and Replacement Algorithm for Network Attached Optical Jukebox

Xuan Liu, Tijun Lu, and Huibo Jia

Optical Memory National Engineer Research Center,
Tsinghua University, Beijing, China
liu2xuan99@mails.tsinghua.edu.cn

Abstract. To improve accessing efficiency, increase the accessing rate and enhance system performance for the massive information storage system, we present a logical accessing model and a new replacement algorithm for network attached optical jukebox. We consider not only the whole accessing distribution, but also the logical relation between each accessing in this logical accessing model. And our algorithm combines the specialty of LFU algorithm and MQ algorithm, especially considers the prefetch method to improve the hit rate in cache buffer. The results of simulative experiment show that our algorithm's performance is much better than other algorithm's in our accessing model. Furthermore, we can conclude that our accessing model can be also used in processor and web buffer.

1 Introduction

For a data storage system, the storing capacity, access speed, safety, dependability etc. are all very important. The optical jukebox is a kind of way for massive information storage, because of its big capacity, high safety, high credibility, is used more and more widely. The appearance of the Network Attached Optical Jukebox (NAOJ)[1] had put optical jukebox and network storage together.

NAOJ, which combined the storing technique of the optical jukebox system technique and networks commendably, is a kind of near line storage system, not real on-line storage system. It adopted a new scheme that treats the optical jukebox as the equipment of NAS connects into the IP Ethernet. The principle diagram of this system is show in Fig.1.

NAOJ uses high-rate disks as Cache of optical jukebox, its working mechanism is described as following: data which is most probably be accessed in the near future is stored in cache according to certain rules. This will decrease frequent accesses to the optical jukebox itself and increase the whole system's access rate. Data in cache is updated dynamically; it will be replaced, deleted or moved by system according to different requirements. With new data's arrival, system will delete those data in cache that couldn't be accessed for a long time in the future and store new data instead.

Much research has been conducted on buffer cache management. Most buffer cache replacement algorithms were designed for processor, web cache and disk array. Such as LRU (Least Recently Used) algorithm, LFU (Least Frequently Used) algorithm. And base on these basic algorithm, many cache replacement algorithm were proposed. These algorithms include: LRU-K[2], 2Q[3], LRFU[4], EELRU[5], LIRS[6], ARC[7], MQ[8], and so on.

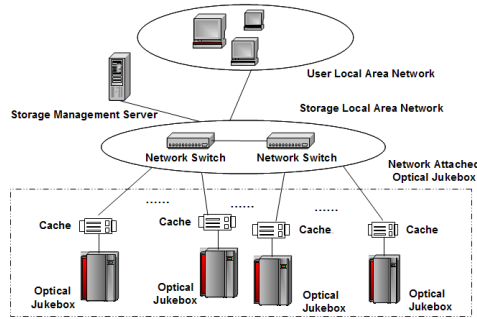


Fig. 1. The principle diagram of Network attached optical jukebox

However, those replacement algorithms could be unavailable in optical jukebox, especially accessing model and replacement algorithm, while the optical jukebox play an important role in the massive storage, so it is necessary to establish a special accessing model and replacement algorithm for optical jukebox.

2 Active Restrict Multi-accessing Model

In general, for one accessing model, there is always a logical relation among several accessing. We can establish corresponding prefetch mechanism through these logical relations. Cache system will prefetch data that may be accessed next time in advance during an access from optical jukebox to save access time. This is a parallel mechanism for store access, and it can save system's access time to a large extent.

Supposing such an access model exists in the NAOJ: previous access determines data to be used in next access. Inspired by the above thing, a new accessing model is proposed.

As in figure 2, this model is called Active Restrict Multi-Accessing Model. This figure is similar to a tree, but not a tree because the relation of each node is uncertain and obscure. In every row of this figure, the number of node means the number of data which may be accessed in current accessing.

From Fig.2, we established an obscure logical relation among several accessing by the active restrict multi-accessing method, which will influence the performance of the whole optical jukebox.

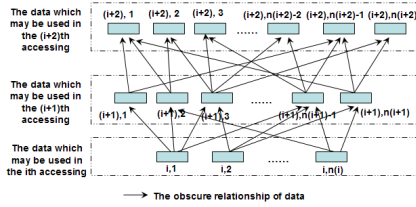


Fig. 2. Active Restrict Multi-Accessing Model

3 Pre-fetch Restrict Algorithm

Based on active restrict multi-accessing model, we propose special replacement algorithm for optical jukebox which is called Pre-Fetch Restrict Algorithm (PFRA). PFRA combines the specialty of LFU algorithm and MQ algorithm, especially considers the prefetch method to improve the hit-rate in Cache.

Fig.3 shows the principle diagram of prefetch restrict algorithm which mainly involves three queues.

There is three queues which is called prefetch queue, hot queue and drop queue. The prefetch queue and hot queue is used to store data while the drop queue is used to store address information of data replaced by hot queue and access frequency information while the actual contents of the data will not be stored. With regard to the whole drop queue, its capacity is by far smaller than those of prefetch queue and hot queue, so we can only consider the capacity of the prefetch queue and hot queue. This is very important.

The process of replacement is to access optical jukebox according to previous access and prefetch the data which has a limitation relation with previous access into prefetch queue. When access arrives, prefetch queue is searched first. Once a data block is hit, it will be moved into hot queue, and the frequency corresponding to the data block is adjusted simultaneously; while prefetch queue is cleared. Then data block is prefetch from optical jukebox according to current access.

Once no data blocks are hit, the hot queue will be searched. If something in hot queue is hit, it will be transferred to I/O port, the frequency of accessing the data block in the hot queue is adjusted, and the sequence of those data blocks will be adjusted according to the access frequency;

Once needful data is not found in the hot queue, the drop queue will be searched;

Once needful data is found in the drop queue, the following operation is similar to the operation of hitting data in hot queue: data is extracted from optical jukebox according to address information stored in the drop queue, then it is transferred to I/O port, the frequency of accessing the data block in the drop queue is adjusted, and the sequence of those data blocks in the drop queue will be adjusted according to access frequency;

Once the needful data is not found in the drop queue, it shows the needful data is not stored in the cache system, and then the optical jukebox has to be accessed. When data is extracted from the jukebox and transferred to I/O port,

the address of the data block needs to be stored simultaneously in the drop queue. Something replaced now is the address of the data in the drop queue; the data's access frequency is the smallest. At the same time the access frequency of the data stored in the drop queue should be set as one.

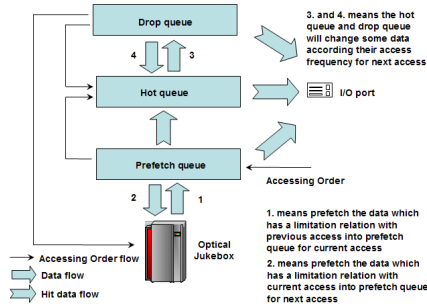


Fig. 3. The principle diagram of Pre-Fetch Restrict Algorithm

There is another important thing that after every access, the hot queue and the drop queue will change some data according to their access frequency for next access.

All the things above are Prefetch Restrict Algorithm.

4 Experimental Results

This section describes the trace-driven simulation of page buffering and the results obtained from our analysis. A program was written to simulate the buffer manager.

During the process of algorithm simulation, considering the characteristics of optical jukebox, we suppose that data in each row of the matrix is arranged according to the size of access probability. Furthermore, the other important function of this hypothesis is predigesting the complexity of experiment.

Once the first data in previous row is hit, the access probability of next row starts with the first data, and it decreases sequentially; once the second data in previous row is hit, the access probability of next row starts with the second data, and it also decreases sequentially; and so on and so forth.

If the accessing probability of B is zero when next row arrange according data A, that is to say, if A is accessed by current accessing, B won't be accessed in next accessing. The relation is shown as figure 4.

This model provides a standard for prefetch. During the prefetch process, we just need to put a few data blocks of which the access probability are biggest into the prefetch queue.

The drop queue's capacitance may be ignored as mentioned in the preamble, therefore, we firstly suppose that prefetch queue and hot queue occupy half of the cache capacitance respectively, and then the simulation results are shown as figure 5.

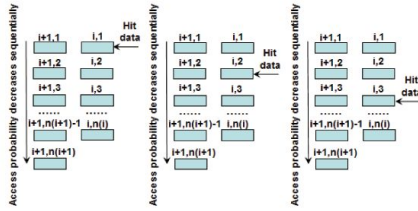
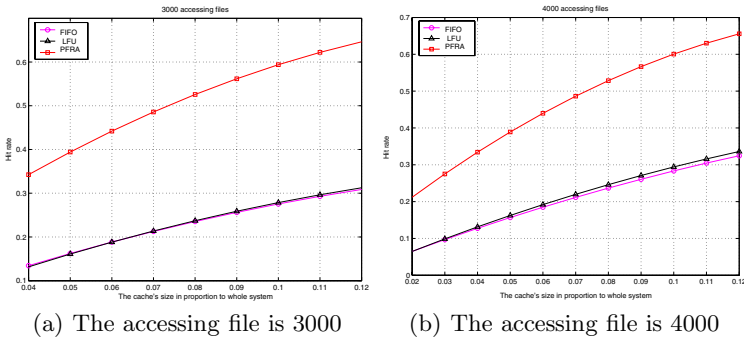


Fig. 4. The property of the simulating matrix



(a) The accessing file is 3000

(b) The accessing file is 4000

Fig. 5. The simulation result

From figure 5, we can make these useful conclusions:

Firstly, we can see that PFRA has provided substantial improvement over FIFO and LFU. Compare with LFU and FIFO, the experiments show that based on the logic relationship among the accessing, PFRA has improved hit rate greatly. Particularly, the hit rate of the prefetch queue makes a great impact on the whole hit rate.

Secondly, hit rate increase with the augment of the cache's size. Obviously, if the cache's size trends the capacity of optical jukebox, the hit rate will tend to be 100 percent, however, in this kind of case, it is insignificant to add a cache to the optical jukebox.

Thirdly, in our accessing model, when 10 percent to 20 percent of the whole system capacity is cache, the hit rate is receivable; and this situation is different from the cache used in the disk array.

5 Conclusion

In this paper we introduced a new replacement algorithm named PFRA. Results from performance evaluation provide evidence that PFRA has significant

performance advantages over conventional algorithms for network attached optical jukebox. Unlike previous replacement algorithms that ignore the logic relationship among the accessing in their replacement decisions, PFRA pays much attention of this relationship, and according to this relationship, the prefetch queue of PFRA plays an important role in the whole replacement process.

The cache model for network attached optical jukebox is also provided. This model includes a special relationship among the accessing. This whole relationship is similar to a tree, but the relation of each node is uncertain and obscure, so we also call this model as quasi tree model.

References

1. HeNing. Studies on the Applications of Network Attached Jukebox in Mass Storage System. [Ph.D.Dissertation].BeiJing, Tsinghua University, (2004)
2. Elizabeth J. O'Neil, Patrick E. O'Neill, Gerhard Weikum. The LRU-K page replacement algorithm for database disk buffering, Proceedings of the 1993 ACM SIGMOD international conference on Management of data, (1993),Volume 22,297-306
3. Johnson T. and Shasha D. 2Q: A Low Overhead High Performance Buffer Management Replacement Algorithm,Proceedings of the 20th VLDB Conference, 1994
4. Donghee L.,Jongmoo C.,Kim J.H.,et al.On the Existence of a Spectrum of Policies that Subsumes the Least Recently Used (LRU) and Least Frequently Used (LFU) Policies, Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, 1999, Volume 27, 134 - 143
5. Smaragdakis Y., Kaplan S. and Wilson P. The EELRU adaptive replacement algorithm, Performance Evaluation, 2003, July, Volume 53, Issue: 2, 93-123
6. Jiang S. and Zhang X.D. LIRS: An Efficient Low Interference Recency Set Replacement Policy to Improve Buffer Cache Performance, Proc. SIGMETRICS, 2002
7. Megiddo N. and Modha D.S. Arc: A Self-Tuning, Low Overhead Replacement Cache, Proceedings of Second USENIX Conference, File and Storage Technologies, 2003.
8. Zhou Y.Y., Chen Z.F and Li K. Second-Level Buffer Cache Management, IEEE Transactions on Parallel and Distributed Systems, Volume. 15, NO. 6, JUNE 2004

Multiple Watermarking Using Hadamard Transform

Haifeng Li, Shuxun Wang, Weiwei Song, and Quan Wen

College of Communications and Engineering, Jilin University, Changchun, 130025, China
lhfvip_2000@163.com

Abstract. In this paper, a multiple watermarking algorithm is proposed which is based on Hadamard transform. Readable watermarks are inserted into Hadamard transform coefficients using quantization. The pseudo-radon permutation is used to enhance the security and robustness of the watermarks. Experimental results showed that the proposed algorithm is invisible and robust to signal processing and geometrical attacks.

1 Introduction

With the development and mature of the Internet and wireless communication techniques the copy and distribution of the digital production becomes easier and faster. The copyright protection is an urgent problem to be resolved. The watermarking technique is considered as a promising method for copyright protection[1]. Most of present watermarking algorithms embed only one watermark, however one watermark is not sufficient under some circumstances.

The reports about multiple watermarks scheme are rather few. Cox et al. [2] extend the single watermark algorithm to embed multiple orthogonal watermarks. The disadvantage is that the original image is needed at the watermark detector, and the watermark capacity is small. Stankovic et al. [3] proposed a scheme utilizing the two-dimensional Radon–Wigner distribution. The lack is that the watermark capacity is small and we cannot judge the validity of the extracted watermark directly. Tao et al. [4] present a multiple watermark algorithm in the DWT domain. The watermark is embedded into all frequencies of DWT. The shortage is that the algorithm is not blind.

This paper proposed a novel image multiple watermarks algorithm. The watermark extraction is blind; The embedding modes are very flexible and we can embed multiple watermarks at the same time or embed different watermark at the different time. The experimental results indicate that our algorithm can give good visual quality and robust to signal processing operations and geometrical distortions.

2 The Hadamard Transform

Given the image $g(x, y)$ of size $N_x \times N_y$, two-dimension HT is expressed as

$$G_{xy}(\mu, \nu) = \sum_{y=0}^{N_y-1} \sum_{x=0}^{N_x-1} g(x, y) \cdot (-1)^{\sum_{r=0}^{p_x-1} x_r \mu_r + \sum_{s=0}^{p_y-1} y_s \nu_s} \quad (1)$$

$$g(x, y) = \frac{1}{N_x} \frac{1}{N_y} \sum_{v=0}^{N_y-1} \sum_{\mu=0}^{N_x-1} G_{xy}(\mu, v) \cdot (-1)^{\sum_{r=0}^{p_x-1} x_r \mu_r + \sum_{s=0}^{p_y-1} y_s v_s} \tag{2}$$

where $G_{xy}(\mu, v)$ denotes the transformation coefficients, $N_x = 2^{p_x}$, $N_y = 2^{p_y}$ (p_x, p_y are the positive integers), $x, \mu = 0, 1, 2, \dots, N_x - 1$, x_r, μ_r respectively are the r th number of the binary codes of x, μ ; $y, v = 0, 1, 2, \dots, N_y - 1$, y_s, v_s are respectively the s th number of the binary codes of y, v .

3 The Watermark Embedding and Extraction Algorithm

3.1 Preprocessing of the Watermark

Binary images are used as the watermarks. We apply the Fibonacci transform to the watermark, and the correlation among the bits is broken to strengthen the robustness[5]. Scanning the permuted watermark in row order and we can obtain a 1D sequence. We adopt the simple bit expanding method to increase the watermark robustness: if the watermark bit is 1, it is coded the sequence $W_{one} = \{1, 1, \dots, 1\}$, where the bit expanding length is N ; if the watermark bit is 0, it will be coded the sequence $W_{zero} = \{0, 0, \dots, 0\}$, where the bit expanding length is also N .

3.2 Watermark Embedding

The detailed embedding approach is described as following:

Step 1. Dividing the original image $g(x, y)$ into non-overlapped blocks of size $B \times B$, which denote $g_s(x', y')$, $s = 0, 1, \dots, S - 1$. That is $g(x, y) = \bigcup_{s=0}^{S-1} g_s(x', y')$, $0 \leq x', y' \leq B$. Applying HT to each block, we can obtain

$$G_s(\mu', v') = HT\{g_s(x', y'), 0 \leq x', y' \leq B\}, 0 \leq \mu', v' \leq B \tag{3}$$

The sub-blocks are pseudo-randomly selected for watermark embedding under control of the key K_1 in order to increase the security of the watermarking system.

Step 2. Scanning the AC coefficients of the selected sub-blocks in Zigzag order and we can obtain a 1D-sequence. As the low and middle frequency components survive in most occasions, we select these as the characteristic collection

$C_s = \{C_s(r)\}$, where $r \in (T_{\min}, T_{\min} + 1, \dots, T_{\max})$, $1 < T_{\min} < T_{\max} < B \times B$, s denotes the s th sub-block.

Step 3. Separating C_k into some regions of the length N . The number of regions can be computed as $H = \left\lfloor \frac{T_{\max} - T_{\min}}{N} \right\rfloor$. Let's the i th region of the s th sub-block as R_s^i , $R_s^i = [T_{\min} + (i-1) \cdot N, T_{\min} + i \cdot N - 1]$.

Step 4. Selecting the region of each sub-block as the embedding position of the j th watermark bit by the key K_W^j . The sequence generated by the key K_W^j is represented as $I^j = \{I^j(s), s = 0, 1, \dots, S - 1\}$, where $I^j(s) \in \{R_s^i, i = 0, 1, \dots, H\}$, $j = 1, 2, \dots, J$, J is the watermark capacity to be embedded.

Step 5. Embedding one watermark bit coded in each region. The detailed embedding rules are as following:

If $W^j(s) = 1$,

$$C'_{ki}(r) = \begin{cases} \text{round}\left(\frac{C_{ki}(r) + 0.5 \times Q_i}{Q_i}\right) \times Q_i - 0.5 \times Q_i & \text{if coded } 1 \\ \text{round}\left(\frac{C_{ki}(r)}{Q_i}\right) \times Q_i & \text{if coded } 0 \end{cases} \quad (4)$$

If $W^j(s) = 0$,

$$C'_{ki}(r) = \begin{cases} \text{round}\left(\frac{C_{ki}(r) + 0.5 \times Q_i}{Q_i}\right) \times Q_i + 0.5 \times Q_i & \text{if coded } 1 \\ \text{round}\left(\frac{C_{ki}(r)}{Q_i}\right) \times Q_i & \text{if coded } 0 \end{cases} \quad (5)$$

where $W^j(s)$ is the s th bit of the un-coded j th watermark, Q_i is the quantization step of the i th region ($1 \leq i \leq H$).

Step 6. Performing inverse block Hadamard transform and the watermarked image $g'(x, y)$ is obtained,

$$g'(x, y) = \bigcup_{s=0}^{S-1} IHT\{G'_s(\mu', v')\} \quad (6)$$

where IHT denotes inverse Hadamard transform.

3.3 Watermark Extraction and Detection

The watermark extraction does not need the original image. First applying block HT to the suspicious image. Second choosing the sub-blocks by the key K_1 , then selecting the watermark embedding characteristic collection C_k . Similarly selecting the region of each sub-block as the extraction position of the j th watermark bit,

$$\hat{C}'_{si}(r) = 0.5 \times (1 - (-1)^{C'_{si}(r) \times 2 / Q_i}) \tag{7}$$

where Q_i is quantization step of the i th region ($1 \leq i \leq H$), $r \in (T_{\min}, T_{\min} + 1, \dots, T_{\max})$, $1 < T_{\min} < T_{\max} < B \times B$, s denotes the s th sub-block.

In succession we separately compute the similarities between the extracted sequence \hat{C}'_{si} and the coded sequences W_{zero} , W_{one} of the watermark 0, 1.

$$Sim_{one} = \frac{\sum_{n=0}^{N-1} W_{one}(n) \oplus \hat{C}'_{si}(n)}{N} \tag{8}$$

$$Sim_{zero} = \frac{\sum_{n=0}^{N-1} W_{zero}(n) \oplus \hat{C}'_{si}(n)}{N} \tag{9}$$

where \oplus denotes the xor operation.

Correspondingly the s th bit $\hat{W}^j(s)$ of the j th watermark is

$$\hat{W}^j(s) = \begin{cases} 1 & \text{if } Sim_{one} > Sim_{zero} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

We map the 1D watermark into 2D matrix, then perform the inverse Arnold transform. The obtained meaningful watermark \hat{W}^j is obtained, which can be judged by human eyes. For simplicity, we adopt the Normalize Coefficient (NC) to evaluate the visual quality of the extracted watermark[6]. If $NC(W^j, \hat{W}^j) > Th$, the watermark W^j is considered to be present; otherwise, the watermark W^j does not exist, where Th denotes the detection threshold.

4 Experimental Results

We test the proposed algorithm on the gray Lena image of 256×256×8 bit. Three watermarks are the 32×32 binary image. Table 1 shows the watermarked image

respectively with one, two and three watermarks. Results show that there are no perceptually visible degradations on the watermarked images.

Table 1. The original image and watermarked images




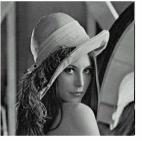
Original image	With one watermark	With two watermarks	With three watermarks
			
/	41.2365 dB	39.1456 dB	37.9587 dB

Table 2. The results after various attacks

Attack types	Extracted watermarks			NC		
	A	B	C	A	B	C
Salt & pepper noise (density 0.005)				0.695	0.701	0.725
Gamma correction (0.6)				0.813	0.778	0.859
Row/column removing (10rows 8columns)				0.851	0.867	0.884
Aspect ratio changing (Length 1.2, Width 1.1)				0.860	0.929	0.923
Rotation (0.1)				1.000	1.000	1.000
Cropping (20%)				0.713	0.691	0.731
Scaling (1.8)				0.920	0.947	0.941

Series of attacks are performed to demonstrate the robustness of our presented algorithm. Table 2 contains the extracted watermarks and the NCs for each of the attacks. The extracted watermarks after these attacks can clearly be identified.

5 Conclusion

This paper proposed a robust blind multiple image watermarking algorithm based on the Hadamard transform. Our algorithm chooses the block HT coefficients as the embedding positions, and we fulfill the multiple watermarks embedding under control of several keys. The watermarks extraction can be implemented without the original image. Experimental results showed that the watermarked image has good visual quality and survives common signal processing and geometrical distortions.

References

1. F. Hartung and M. Kutter, Multimedia watermarking techniques[C], Proceedings of the IEEE, 1999, 87(7): 1079-1107.
2. I. J. Cox et al. Secure spread spectrum watermarking for multimedia[J]. IEEE Trans. on Image Processing, 1997,6(12): 1673-1687.
3. Srdjan Stankovic, Igor Djurovic, and Ioannis Pitas, Watermarking in the space/spatial-frequency domain using two-dimensional Radon–Wigner distribution[J]. IEEE Trans. on Image Processing, 2001, 10(4): 650-658.
4. Peining Tao and A. M. Eskicioglu. A robust multiple watermarking scheme in the discrete wavelet transform domain[C]. Optics East 2004 Symposium, Internet Multimedia Management Systems V Conference, 2004.
5. Haifeng Li, et al. A Novel Watermarking Algorithm Based on SVD and Zernike Moments[C]. 2005, LNCS 3495, 448-453.
6. Miller M. L, Bloom J A. Computing the probability of false watermark detection[A]. Proceedings of 3rd International Workshop Information Hiding[C], Germany, 1999: 146-158.

An Immunity-Based Intrusion Detection Solution for Database Systems

Ke Chen, Gang Chen, and Jinxiang Dong

College of Computer Science, Zhejiang University, Hangzhou, P.R.China 310027
ar_ke@hotmail.com, cg@cs.zju.edu.cn, djx@cs.zju.edu.cn

Abstract. Database intrusion detection has been an important research area in database security. It focuses on malicious transaction attacks, which cannot be handled by traditional database security mechanisms, such as authorization, access control, integrity control, and so on. Although there have appeared some intrusion detection systems, current researches on malicious transaction detection are limited in accuracy and efficiency. Inspired by natural immune system, we propose a novel immunity-based intrusion detection solution for database system in this paper. It provides an additional layer of defense against DBMS misuse, especially malicious transactions. The ability to learn and to adapt to the environment dynamically entitles the system to detect both known and unknown malicious transaction intrusions efficiently. Simulations show that the database intrusion detection system based on data immunity can accelerate detection of malicious transaction attacks and improve its accuracy without causing any other performance penalty.

1 Introduction

With the rapid development of computer and communication technology, database security has received much attention. However, almost all the traditional database security methods cannot distinguish legitimate transactions from malicious ones [1,2]. As a result, here come the database intrusion detection techniques as a critical means in handling this kind of intrusions.

The main goals of database intrusion detection include: detecting malicious transactions before their being committed, then dropping and rolling back these transactions; otherwise when malicious transactions have been committed and have caused damages, locating the damaged parts and repairing them as soon as possible, so the database can still provide services during the attacks. In these years, the techniques have received much attention and made great progresses, such as: Chung et. al. [2] presented an intrusion detection system for inner misuse; Rakesh et. al. [3] extended the function of DBMS, recorded and analyzed the data access habit of every user and compared real-time whether the every access accords with the traditions; Lee et. al. [4] studied and utilized update frequency of every data object in the database to detect intrusions. Peng L [5] introduced an intrusion-tolerant database system, which can operate through attacks in such a way that the system can continue delivering essential services in the face of attacks.

As a whole, current researches about intrusion detection for database systems have applied mature techniques similar to the intrusion detection in computer networks, but are limited in detecting, isolating and repairing damages caused by malicious transactions. Although a lot of anomaly detection algorithms [6] have been proposed recently, they are difficult to be directly applied to the malicious transaction detection due to its special characteristics.

In this paper, we novelly construct a model of Intrusion Detection System referring to digital immunity, and organically combine malicious transaction detecting, pre-alarmed, isolating, and damage repairing together. Moreover, based on rapid malicious transaction detection, the immune system can produce antibodies through immune analyzing, and then sent them out to all the modules respectively to enhance resistance of the whole database system.

2 Intrusion Detection Model for Database Based on Digital Immunity

2.1 Definitions

Definition 1. Antibody

The antibody of malicious transaction m is defined by a quadruple $a_m = (O, PreC, Act, R)$, where:

- 1) O is a set of objects which antibody would be applied on. In this model, the objects include the outer pre-alarms and the modules of the inner database server.
- 2) $PreC$ is the precondition of antibody action, which is a condition expression. When it is satisfied, antibodies will be activated and take actions.
- 3) Act is the immune action set executed when the antibody is activated.
- 4) R is an action restriction set that must be satisfied during the course of immune action being executed.

Definition 2. Detection Immunocyte

Detection immunocytes with antibody are used to detect and identify malicious transactions. Let B be the detection immunocyte set for malicious transactions and $b \in B$ such that $b = (a_m, num, age)$ is a detection cell for the malicious transaction m , where:

- 1) a_m is the antibody carried by the detection immunocyte.
- 2) num is the time when the antibody in the detection immunocyte is activated.
- 3) age is the age of the detection immunocyte, if a_m has not been activated by far when age is beyond a certain threshold ζ , then the detection immunocyte will be discarded.

Detection immunocytes are classified into Memory Immunocytes, Mature Immunocytes and Immature Immunocytes, where:

- 1) $B_m = \{x \mid x \in B, x.num \geq \lambda\}$ denotes Memory Immunocyte set, where, λ is the threshold of the activated time.

- 2) B_p , the Mature Immunocytes, is composed of the detection cells, which endure self-tolerance (cannot be activated by legitimate transactions) and whose activated times are not beyond the threshold λ .
- 3) B_i , the Immature Immunocytes, is composed of the detection cells, which do not endure self-tolerance and are produced randomly.

To guarantee the detection rate in the model, the sum of the Mature and Immature immunocytes is a constant, that is, $|B_i| + |B_p| = \theta$.

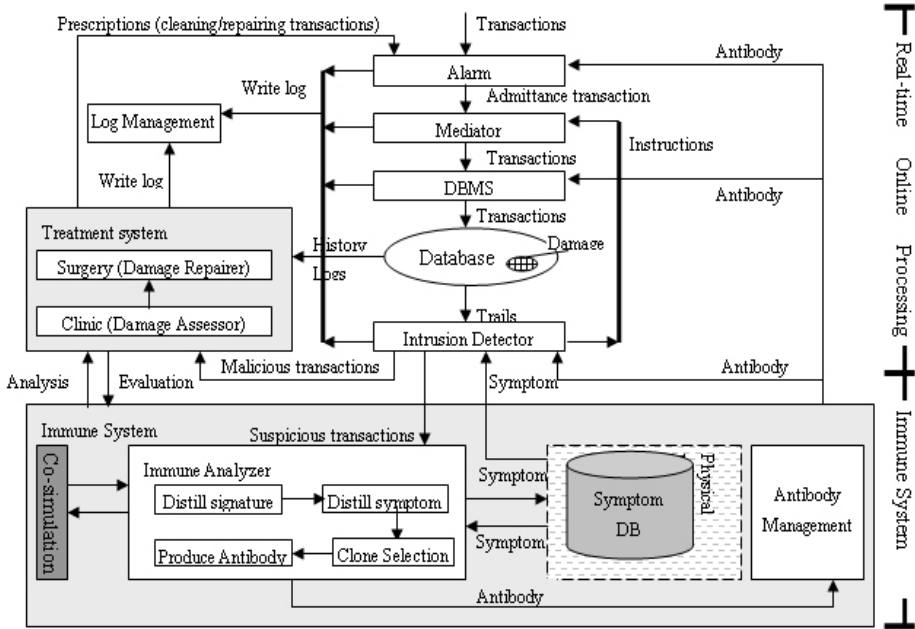
Definition 3. False Negative Rate α /False Positive Rate β

False Negative Rate α is the probability of the omission of malicious transactions. False Positive Rate β is the probability of treating legitimate transactions as malicious ones.

2.2 The Model

Figure 1 shows the architecture of intrusion detection model for a database system based on digital immunity. According to the strict real-time requirements of a database transaction processing, the system is divided into Real-time Online Processing System (RTOPS) and Immune Analysis System (IAS). And the RTOPS is composed of Alarm, Mediator, Intrusion Detector, and Treatment System. With the help of antibodies produced by Immune Analyzer and traditional intrusion detection techniques for the database, the RTOPS takes charge of most malicious transaction intrusions. We describe these modules in the following:

- A. *Alarm* functions like the human skin. Alarms are deployed in every database client in the network. They act as the first gateway against malicious transaction intrusions. With the help of antibodies carried by the memory immunocytes, the Alarm can detect and quickly respond to most malicious transaction intrusions, especially those previously encountered. The antibodies sent to the Alarm may even influence the security policy of the system. For instance, the Alarm can automatically reject any transaction request from the user who has sent out the malicious transactions before, until the restrictions are deleted manually by the security manager. With the time going by, the Alarm can work more effectively if more memory immunocytes are in the system.
- B. The function of the *Mediator* is to adjust the number of transactions submitted to the system automatically, based on alarm numbers, alarm level, damage assessment, and current workload. In face of intense attacks, the Mediator adaptively slows down or even suspends the execution of new transactions, which can help stabilize the data integrity level and prevent damages from being spread seriously.
- C. The *Intrusion Detector*, cooperating with the traditional intrusion detection for a database system, monitors and analyzes the trails of the database sessions and transactions in a real-time manner, and identifies malicious transactions as soon as possible according to the symptoms preserved in the Symptom DB. It also informs Mediator the number of malicious transactions and alarm level, and sends the suspicious and malicious transactions to Immune Analyzer for further distinguishing and analyzing at the same time.



D. The *Treatment system* consists of the Surgery and the Clinic. The system input includes analyzed data from the immune system, alarm messages from the Intrusion Detector, the logging data and auditing data of the DBMS. The Clinic takes charge of evaluating the severity and the extent of the damages caused by malicious transactions. It locates damaged data objects and finds out transactions related to these data objects directly or indirectly. The Surgery operates on the database according to the evaluating data produced by the damage evaluator. It generates prescriptions and executes transaction cleaning procedures to cure the damaged database.

The *Immune System* is responsible for producing and managing the immunocytes for malicious transactions. In fact, it is not required to be as “real-time” as the RTOPS is. As the core of the intrusion detection system for a database system, the Immune system consists of Immune Analyzer, Symptom DB and Antibody Management etc. The *Immune Analyzer* continually produces immature immunocytes. These immunocytes suffer a period of tolerance time called self-tolerance upgrade to turn into mature immunocytes. The immature immunocytes that fail to upgrade themselves will be deleted. To guarantee the availability of the system, the sum of the numbers of immature immunocytes and matures is a constant. Based on Clone Selection Algorithm, the symptoms would be written into *Symptom DB* that is physically isolated when the mature immunocytes succeed in matching with the suspicious transaction. Different kinds of antibodies are produced and sent to modules in the system by the *Antibody Management*. Mature immunocytes will be upgraded to memory immunocytes when the number of the matches between mature immunocyte’s and malicious transactions reaches a threshold. Also, if there is no match at all after a period of time, these ma-

ture immunocytes will be deleted. Manual determination may be required to realize *Co-stimulation* in the system.

In this model, antibodies are delivered automatically to every database client in the network, or even to every server of a distributed database system. There are two kinds of antibodies in the model:

- a) Antibody I: antibodies that are sent to the Alarm, carried by the memory immunocytes, and mainly responsible for rejecting malicious transaction intrusion quickly at the entrance to the database. Their immune action is usually directly discarded.
- b) Antibody II: antibodies that are sent to the DBMS, carried by the mature immunocytes, are responsible for analyzing transaction process flows in real-time. After their being activated, their immune actions include rolling back the malicious transactions and automatically modifying security policy about the DBMS, i.e., locking some tables.

3 Simulations

Supported by the National High-Tech Research and Development Plan of China, a prototype system is developed on Oscar DBMS by our research team, and the simulations are also conducted.

The hardware environment of our simulations is CPU: PIV 2.4G, main memory: 512M, hard disk: 40G, network bandwidth 100 MBPS, and operating system is Windows2000 Advanced Server. In the simulations, clients commit 100 transactions and half of them are malicious transactions every second. Simulation system maintains a legitimate transaction set and a malicious transaction set, both of which have 50 elements at first. And the committed transactions come from these two sets, with a few variances or no change at all. Due to the limitation of paper length, we only discuss classic simulations here.

Figure 2 shows the influence of the sum of non-memory immunocytes θ on the false rate. We can see from the figure, with the increasing of θ , that there appear more memory immunocytes. With the increasing of the false positive rate, the false negative rate goes down. Figure 2 also indicates the false rate is strongly affected by the sum of non-memory immunocytes.

Figure 3 illustrates the relationship between the false rate and the threshold of the activated time λ . As we can see, the bigger the λ is, the harder the mature immunocytes can evolve into memory immunocytes, which causes the reduction of memory immunocytes in the system. Consequentially the false negative rate increases, while the false positive rate changes oppositely. We also note that, compared with θ in figure 2, λ has a lower impact on the false rate.

In our simulations, when system has been mature, alarms can detect more than 95% malicious transactions, and the time spent is negligible. The performance losing is not bigger than 5%. And with the increasing of system running time, the losing decreases. The mean latency of malicious detection completely meets the requirements.

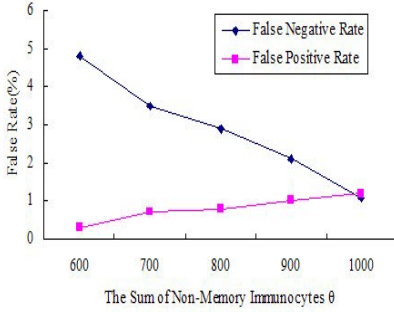


Fig. 2. False rate changes with θ

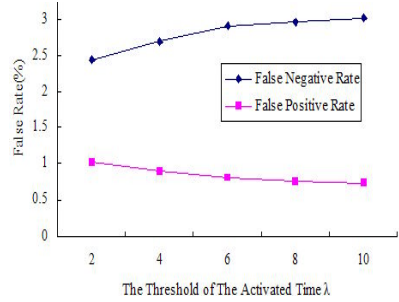


Fig. 3. False rate changes with λ

4 Conclusions

In this paper, we focus on transaction-level intrusion detection, which is based on the fact that most attacks are from insiders. Simulations show that the database intrusion detection system based on data immunity can be very effective against the malicious transaction attacks. Moreover, existing low-level intrusion detection mechanisms can be seamlessly and easily integrated into our solution to build a multi-layer, intrusion-tolerant database recovery system.

References

1. Paul A, Sushil J, Peng L. Recovery from malicious transactions. *IEEE transactions on knowledge and data engineering*, 14(5): (2002)1167-1185
2. Chung C.Y, Gertz M, Levitt K. Demids: a misuse detection system for database systems. In: *Proc. of Third International IFIP TC-11 WG11.5 Working Conference on integrity and Internal Control in Information Systems*, Amsterdam: Kluwer Academic Publishers. (1999)159-178
3. Rakesh A, Ramakrishnan S. Fast algorithms for mining association rules. In: *Proc. of 20th International Conference on Very Large Data Bases*. Berlin: Morgan Kaufmann,. (1994)487-499
4. Lee W, Xiang D. Information-theoretic measures for anomaly detection. In: *Proc. 2001 IEEE Symposium on Security and Privacy*. Oakland: IEEE Computer Society, (2001)130-143
5. Peng L. DAIS: A real-time data attack isolation system for commercial database applications. *17th Annual Computer Security Applications Conference (ACSAC'01)*. New Orleans: IEEE Press, (2001) 219-229
6. Rao X, Dong CX, Yang SQ. An intrusion detection system based on support vector machine. *Journal of Software*, 14(4): (2001)798-803(in Chinese with English abstract)

Filtering Duplicate Items over Distributed Data Streams

Tian Xia¹, Cheqing Jin^{1,2}, Xiaofang Zhou³, and Aoying Zhou¹

¹ Department of Computer Science and Engineering,
Fudan University Shanghai, 200433, P.R. China
{txia, cqjin, ayzhou}@fudan.edu.cn

² East China University of Science and Technology,
Shanghai, 200237, P.R. China
cqjin@ecust.edu.cn

³ School of Information Technology and Electrical Engineering,
The University of Queensland, Brisbane QLD 4072 Australia
zxf@itee.uq.edu.au

Abstract. In recent years many real time applications need to handle data streams. We consider the distributed environments in which remote data sources keep on collecting data from real world or from other data sources, and continuously push the data to a central stream processor. In these kinds of environments, significant communication is induced by the transmitting of rapid, high-volume and time-varying data streams. At the same time, the computing overhead at the central processor is also incurred. In this paper, we develop a novel filter approach, called **DTFilter** approach, for evaluating the windowed distinct queries in such a distributed system. **DTFilter** approach is based on the searching algorithm using a data structure of two height-balanced trees, and it avoids transmitting duplicate items in data streams, thus lots of network resources are saved. In addition, theoretical analysis of the time spent in performing the search, and of the amount of memory needed is provided. Extensive experiments also show that **DTFilter** approach owns high performance.

1 Introduction

Recently, a few *continuous query* systems have been proposed to cope with applications involving continuous data streams[1,3,4]. The rate of a stream is very rapid and the volume is unbounded. Motivational applications include stock tickers, network monitoring, telecom call records, sensor networks, and many others. Lots of applications prefer to sliding window model where only a collection of latest N elements are considered. Distributed processing is another key topic in stream applications. In distributed environments, each stream is observed and summarized by its party and the resulting synopses are then collected at a central site. Obviously, the greatest challenge is how to reduce the communication volume.

One of the most fundamental problems over data stream is how to filter elements to reduce communication amount. Although a few literatures have been focused on build filters for queries like *count*, *distinct count*, *min*, *max*, work on filtering duplicate elements over distributed data streams is still rare. However, it meets some importance. Consider a scenario where same events are detected by one remote site in a short period of time. It is better to eliminate duplicate events to save bandwidth.

In this paper, we propose a **DTFilter** approach to filter duplicate elements. The core structure of **DTFilter** approach is two height-balance trees deployed in each remote site, indexed by the element's id or arrival time. Experimental results show that **DTFilter** approach can significantly reduce the communication volume among sites.

We introduce some preliminary knowledge in Section 2. In Section 3, our novel **DTFilter** approach is described in detail, followed which experimental results are reported in Section 4. Finally, a brief conclusion is presented in Section 5.

2 Preliminary Knowledge

2.1 Problem Definition

We consider a distributed environment where remote sites continuously generate a stream of data. Once a continuous query has been registered, a remote site may send querying results to central processor when seeing a newly arrival tuple. Consider a SQL-like statement "SELECT DISTINCT $S.id$ FROM S WINDOW W ", where S is a stream containing at least two fields, *id* and *time*. Field *id* presents one characteristic of the tuple; *time* presents the time when tuple arrives. This query transfers distinct elements in most recent W time unit.

Example 1. Assume the first ten tuples of a stream is " $x, f, d, x, c, f, f, b, a, y$ ". When the window size $W = 4$, tuples at time 4, 7 don't need to be transferred to central site due to duplication elimination.

2.2 Related Work

Recently, research work on data streams appears to be a hot topic in database fields. Good survey papers include [1,4].

A few literatures are focused on building filters for stream systems. In [8], Olston et al. proposed a filter in distributed stream environment to ensure a small error bound in the central site. Aware that Kalman Filter can predict element's value after a few updates, Jain et al.[5] proposed an adaptive filtering strategy by fully using Kalman Filter. In *Predictive Filtering* approach[6], matching predictors are deployed both at remote sites and central site. Only when the difference between the actual and predicated values at the remote site exceeds a threshold, the update is streamed.

However, above filters aren't focused on duplication elimination. Although *bloom filters*[2,7] can support queries like: "Is element x in a set S ?", this technique is hard to adapt to windowed model.

3 DTFilter Approach

In this section, we begin to depict our solution. Obviously, the problem can be solved by a naive method, in which an array is allocated to reserve tuples in stream. When seeing a newly arrival tuple at time t , the method begins to search the buffer to check whether a tuple with same id has arrived in last W time unit. If a tuple happens to be found, nothing else requires to do. Otherwise, the new tuple must be sent to the central site after being reserved in the array. However, besides the simplicity, this naive method is expensive in searching a right tuple. In the worst case, nearly all tuples in the array are required to examine.

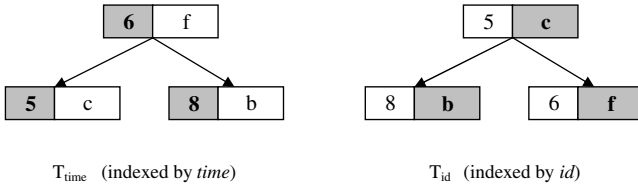


Fig. 1. An example of DT structure

Here, we propose a novel approach, called **DTFilter** (Dual Tree Filter) approach, to cope with the problem. The core structure of **DTFilter** is two height-balanced trees (T_{id} and T_{time}) deployed on each remote site. Both T_{id} and T_{time} are indexes for a same set of tuples coming from input stream, while T_{id} is indexed by field id and T_{time} is indexed by field $time$. **Fig. 1** demonstrates an example of DT structure of **Example 1** at time 8, when the past 4 tuples in format $(time, id)$ are $(5, c), (6, f), (7, f)$ and $(8, d)$. Tuple $(7, f)$ can be eliminated because its id is same with the previous tuple. In T_{time} , three tuples are indexed by $time$, while in T_{id} tuples are indexed by id . We can observe that T_{id} and T_{time} are just indexed for a same tuple set: $\{(5, c), (6, f), (8, d)\}$

Algorithm DTFilter (**Algorithm 1**) shows the way to maintain DT structure when seeing a new tuple q in the stream. Initially, **FindItemInTid** routine is called to find an entry p in T_{id} matching $p.id = q.id$ (at Line 1). Because T_{id} is indexed according to field id , this step can be done quickly. Once no such entry can be found from T_{id} , we can decide that no tuple with id equal to $q.id$ has arrived in period $(t - W, t]$, so that q must be sent to the central site (at Line 3). Furthermore, T_{id} and T_{time} are to be maintained for future processing. Firstly, check whether there is free buffer to be used. If no free space is left for the approach, the oldest tuple must be removed from T_{id} and T_{time} at the same time. Secondly, tuple q is inserted into T_{id} and T_{time} (at Lines 4-5).

Once a tuple is found according to **FindItemInTid** routine, we can decide that a tuple (tuple p) with id equal to $q.id$ has ever arrived. We should continue to check whether p exist in $(t - W, t]$ or not (at Line 7). When $p \in (t - W, t]$, tuple q is not required to sent to central site due to duplication elimination.

Algorithm 1. `DTFilter(Tuple q)`

```

1:  $p = \text{FindItemInTid}(q.id)$ ;
2: if ( $p = \text{NULL}$ ) then
3:   Send  $q$  to central site;
4:   Remove oldest tuple from  $T_{id}$  and  $T_{time}$  if no free buffer exists;
5:   Insert  $q$  into  $T_{id}$  and  $T_{time}$ ;
6: else
7:   if ( $q.time - p.time < t$ ) then
8:     Send  $q$  to central site;
9:      $p' = \text{FindItemInTtime}(p.time)$ ;
10:    Update  $time$  of  $p'$  and  $p$  in  $T_{id}$  and  $T_{time}$ ;
11:   end if
12: end if

```

Otherwise, p is sent to central site, followed which DT structure is revised (at Lines 8-10). We can firstly find the tuple p' in T_{time} matching $p'.time = p.time$ through `FindItemInTtime` routine. Finally, update the $time$ of p and p' in T_{id} and T_{time} .

Performance Analysis

We begin to analyze the performance of `DTFilter`. Because no previous work is focused on this problem, here we mainly compare the performance with the naive method described before.

As analyzed above, the naive method may require to scan all elements in the buffer to check whether a tuple with same id exists or not. In other words, it requires $O(M)$ to complete the task, where M is the buffer size. Contrarily, in `DTFilter`, this kind of cost can be reduced to $O(\log M)$, because T_{id} is built with index on field id . The cost on maintaining DT is similar. For some kinds of height-balanced tree, such as b-tree and red-black tree, the cost on removing one node or adding one node is sublinear to the number of nodes in the tree. It means that the cost can also be $O(\log M)$. As a result, `DTFilter` only requires $O(\log M)$ to process one tuple.

Additionally, the space required in `DTFilter` approach is comparable with the naive method. In the naive method, the memory space is linear to the number of tuples in the buffer, while in `DTFilter` method, the memory space is also linear to the number of tuples reserved.

4 Experimental Evaluations

We report experimental results here. All codes were written in C++ and implemented at an Intel Pentium IV 2.4GHz PC with 1G memory. The red-black tree is used to construct DT structure for it's an excellent height-balance tree. We first test the *costgain* of `DTFilter` approach, which is the ratio of the number

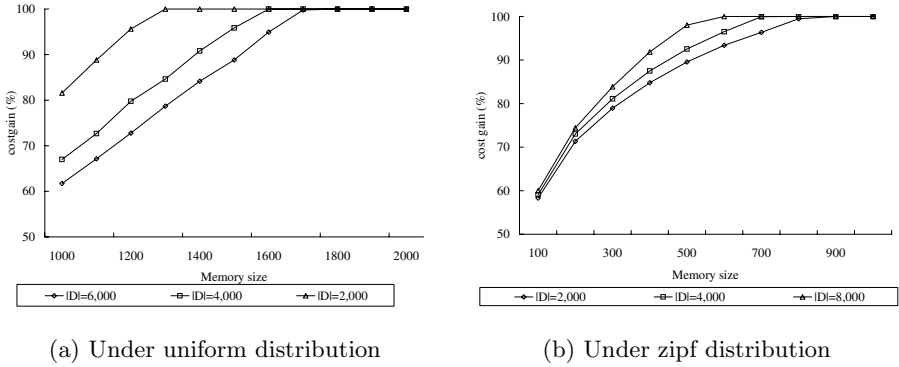


Fig. 2. Cost gain of DTFILTER approach under different distributions

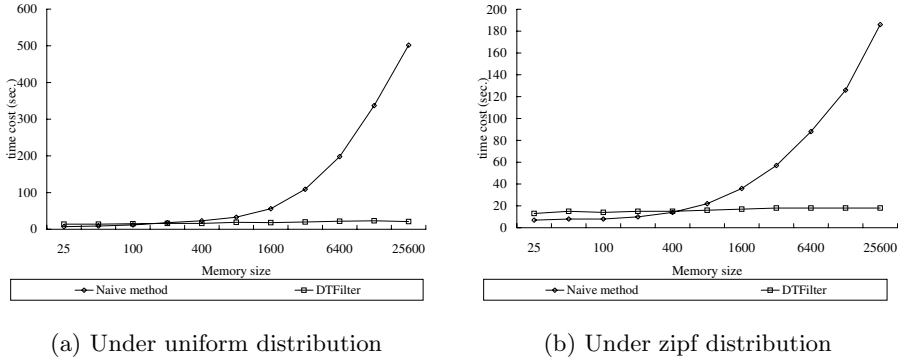


Fig. 3. Time Cost of DTFILTER approach under different distributions

of detected duplication to the real duplications over stream. Then, we continue to compare the time cost to process one incoming tuple with the naive method.

Obviously, *costgain* of DTFILTER approach can reach 100% when given enough memory space. **Fig. 2(a)** shows *costgain* under three synthetic datasets in uniform distribution. The size of three datasets are all 100,000 while the number of distinct elements is set to 2,000, 4,000 and 6,000 respectively. We set the window size W to 2,000 and increase the memory size from 1,000 to 2,000. The x-axis presents the memory size used in the experiment; y-axis presents the value of *costgain*. We can observe that the *costgain* increases if given more memory space. The *costgain* would approach 100% once the memory size is close to W . When given same memory space, the *costgain* can be higher for a dataset with smaller number of distinct elements. **Fig. 2(b)** shows the *costgain* under other three synthetic datasets in zipf distribution with $z = 1$. The number of distinct elements are set to 2,000, 4,000 and 8,000 respectively. Similarly, more memory space would lead to higher *costgain*.

We then test the time cost of **DTFilter** approach. **Fig. 3(a)** shows the time cost under a uniform dataset. The uniform dataset contains 100,000 tuples with cardinality equal to 50,000. We set the window size W to 400. The x-axis presents the memory size used in the experiment; y-axis presents time cost to process tuples. No matter how the memory space is increased, our **DTFilter** approach can still be completed very quickly. Contrarily, the naive method must pay more time to process tuples. In the worst case, **DTFilter** approach can be 20-times faster than the naive method. **Fig. 3(b)** shows the time cost under a zipf distribution containing 100,000 tuples with cardinality equal to 50,000 and $z = 1$. We set the window size W to 100. Similarly, our **DTFilter** approach can be quite faster than the naive method.

5 Conclusion

In this paper, we propose a novel **DTFilter** approach to perform an efficient data filtering on remote sites under distributed environment for duplication elimination. To the best of our knowledge, no similar methods have been done on this topic. The thorough experimental results show that our method has a good performance.

Acknowledgement

The authors would like to thank Dr. Weining Qian for his comments upon this paper, and would like to thank Mr. Jiabin Chen and Mr. Haiyi Chen for implementing experiments.

References

1. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proc. of 2002 ACM Symp. on Principles of Database Systems*, 2002.
2. A. Broder and M. Mitzenmacher. Network applications of bloom filters: A survey. In *Proc. of the 40th Annual Allerton Conference on Communication, Control, and Computing*, pp. 636-646, 2002.
3. D. Carney, U. Cetintemel, M. Cherniack, and C. Convey. Monitoring streams—a new class of data management applications. In *Proc. of VLDB*, 2002.
4. L. Golab and M. T. Özsu. Issues in data stream management. *ACM SIGMOD Record*, 32(2):5–14, 2003.
5. A. Jain, E. Y. Chang, and Y. Wang. Adaptive stream resource management using kalman filters. In *Proc. of SIGMOD*, 2004.
6. V. Kumar, B. F. Cooper, and S. B. Navathe. Predictive filtering: A learning-based approach to data stream filtering. In *Proc. of the 1st workshop on Data Management for Sensor Networks (DMSN'04)*, 2004.
7. M.V.Ramakrishna. Practical performance of bloom filters and parallel free-text searching. *Communications of ACM*, 32(10):1237–1239, October 1989.
8. C. Olston, J. Jiang, and J. Widom. Adaptive filters for continuous queries over distributed data streams. In *Proc. of SIGMOD*, 2003.

An Optimized K-Means Algorithm of Reducing Cluster Intra-dissimilarity for Document Clustering*

Daling Wang, Ge Yu, Yubin Bao, and Meng Zhang

School of Information Science and Engineering, Northeastern University
Shenyang 110004, P.R.China
{dlwang, yuge, baoyb}@mail.neu.edu.cn

Abstract. Due to the high-dimension and sparseness properties of documents, clustering the similar documents together is a tough task. The most popular document clustering method K-Means has the shortcoming of its cluster intra-dissimilarity, i.e. inclining to clustering unrelated documents together. One of the reasons is that all objects (documents) in a cluster produce the same influence to the mean of the cluster. SOM (Self Organizing Map) is a method to reduce the dimension of data and display the data in low dimension space, and it has been applied successfully to clustering of high-dimensional objects. The scalar factor is an important part of SOM. In this paper, an optimized K-Means algorithm is proposed. It introduces the scalar factor from SOM into means during K-Means assignment stage for controlling the influence to the means from new objects. Experiments show that the optimized K-Means algorithm has more *F-Measure* and less *Entropy* of clustering than standard K-Means algorithm, thereby reduces the intra-dissimilarity of clusters effectively.

1 Introduction

Recently, the most popular document clustering method is K-Means[4]. It has such advantages as straightforward processing, easily programming, and the relative low computational complexity. However, due to the high-dimension and sparseness properties of documents, especially the same influence to means from all new objects, K-Means has its unfitness. One of the problems is that after selecting k centers (means) randomly, for each remaining object, the similarity between the objects with the each cluster is computed. The process will result in intra-dissimilarity. The intra-dissimilarity means that the objects grouped into the same cluster are totally unrelated. For example, one cluster mean is represented as $\{T_1, T_2, T_3, T_4, T_5\}$, and there are two documents represented as $\{T_1, T_2, T_3, T_6, T_7\}$ and $\{T_3, T_4, T_5, T_8, T_9\}$. If the similarity is measured by computing distance between an object (document) with all cluster means, both of the two documents will be assigned to the same cluster represented by $\{T_1, T_2, T_3, T_4, T_5\}$. But the two documents are not similar to each other obviously. With the increasing of dimension, the intra-dissimilarity will be so serious that the documents in one cluster are totally unrelated.

* This work is supported by National Natural Science Foundation of China (No. 60173051)

SOM (Self Organizing Map) is a method to reduce the dimension of data and display the data in low dimension space[3], and it has been applied successfully to clustering of high-dimensional objects. The scalar factor is an important part of SOM. For reducing the intra-dissimilarity of clusters, we reference to the scalar factor from SOM. During the assignment step in K-Means, the coming document imposes the mean of one cluster it belongs to, some influence making the mean shift towards the coming document, but the influence will decrease with the coming of documents.

2 Preprocessing and Similarity Definition of Document

Document clustering is defined as follows. Given a document set $D=(d_1,d_2,\dots,d_n)$, under the control of some criterion functions, a cluster set $CS=\{c_1,c_2,\dots,c_k\}$ is obtained, where $c_j \subset D$ ($j=1,2,\dots,k$), and for any $d_i \in D$, there is $d_i \in c_j$ and $c_j \in CS$.

We use BOW (Bag of Word) to represent a document by performing the following steps on the document set. Firstly, all stopwords¹ are removed. Secondly, all the remaining terms are stemmed using Porter Stemmer². Then a term space T is obtained. Based on T , we build a term vector for every document $d \in D$. For each document, the terms $t \in T$ are weighted by $tfidf$ [7]. We select the first n terms with maximum $tfidf(d,t)$ as a n -grams vector to represent document d . The similarity between two documents $d_1, d_2 \in D$ is computed using the cosine of their term vector.

3 Intra-dissimilarity of Cluster

K-Means Algorithm is the most popular document clustering algorithm, standard K-Means algorithm for document clustering works as follows.

Algorithm1: Standard K-Means Algorithm for Document Clustering

Input: The number of clusters k and a document set with n objects

Output: A set of k clusters that minimize the criterion function

Method:

- 1) randomly choose k objects as the initial cluster centers;
- 2) **repeat**
- 3) assign each object to the cluster to which the object is the most similar based on the mean of the objects in the cluster;
- 4) update the cluster means;
- 5) **until** meeting a given criterion function;

According to the steps of standard K-Means algorithm, step 3)~4) will bring on intra-dissimilarity. For example, $D_1=\{T_1, T_2, T_3\}$, $D_2=\{T_1, T_2, T_4\}$, $D_3=\{T_2, T_4, T_5\}$. If computing their similarity using cosine function and selecting D_1 as the mean, here $\cos(D_1, D_2)=\cos(D_1, D_3)=0.5$ is similar enough, but $\cos(D_2, D_3)=0.2$ is too dissimilar. We think that this is due to during assigning a new object to a cluster, the mean of the cluster not change with the new object.

¹ From: http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

² From: http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/porter.c

4 Optimized K-Means Algorithm

4.1 SOM Algorithm

SOM consists of sample data and Map. It first assigns a weight for every neuron in a given Map randomly, and the weight has the same dimensions with the sample. Then it learns the neurons continuously with sample data so that similar objects can be mapped to the same neuron for clustering. The algorithm is shown as follows.

Algorithm2: SOM Algorithm

Input: Map and samples

Output: trained map

Method:

- 1) initialize Map so that every neuron in Map is assigned an initial weight;
- 2) For $i=1$ to n (n is a parameter of given iteration)
- 3) Select a sample randomly;
- 4) search best matching unit (BMU) with the sample in Map;
- 5) adjust BMU and its neighborhood neurons' weight;
- 6) $i++$;
- 7) End for;

The key is Step 5), which concerns the learning process of neural-network, and the process will result in BUM more similar with its neighborhood unit.

4.2 Optimized K-Means Algorithm

It has been mentioned in section 3 that the K-Means might result in the intra-dissimilarity, and we also analyze that the intra-dissimilarity is due to the high-dimension and sparseness properties of documents, especially the influence of new documents. Generally, one document corpus includes many words, and the vector representation of the each document includes at least hundreds of words. But the similarity between two documents might be determined by only 10 percents of the vector or even less. In step 3) of Algorithm1, each document is assigned to one cluster based on its similarity to each cluster mean, the similarity between them lying on only a little part of vectors, which is the key of generating intra-dissimilarity.

In SOM, the coming sample vector finds its best matching unit (BMU) and then imposes a learning function on neighborhoods of the BMU. The leaning function changes the value of neighborhoods based on BMU, making them shift towards BMU. This step is called *Scale Neighbors*. By this means, the SOM makes sure that the samples captured by one neuron are of the most similar.

Inspired by *Scale Neighbors* in SOM, we improve the step 3) in Algorithm1. Different from keeping the mean unchanged, with the documents assigned to the clusters, the corresponding cluster mean is influenced by the coming document, making the mean shift towards the coming document. But if each of the coming documents imposes the same influence on the mean, the new mean can not reduce the intra-dissimilarity or even deteriorate it. So we introduce the scalar factor to control the influence of orderly coming documents, which is defined in formula (1).

result is. The latter is used to evaluate the correlation among objects in a cluster, and the less it is, the better the correlation is. We also test the time cost of the two K-Means algorithms on a computer with 1GHz CPU and 128M memory.

The experiment results are illustrated in Fig.1, Fig.2 and Fig.3, which are the *F-Measure*, *Entropy* and time cost, respectively.

Obviously, the optimized K-Means algorithm has better *F-Measure* and *Entropy* than the standard K-Means algorithm from Fig.1 and Fig.2. Because of introducing scalar factor in clustering, the intra-dissimilarity shortcoming is conquered to a great extent thereby quality of document clustering is improved.

From Fig.3, the time cost of the two K-Means algorithms has not obvious difference, especially when number of documents becomes more. There are two reasons. Firstly, in optimized K-Means algorithm, for each newly assigned document, it requires k more times similarity computation. This process increases the time cost. Secondly, the influence of scalar factor makes that the constringency is quickened. So in the mass, the time cost has not too much change.

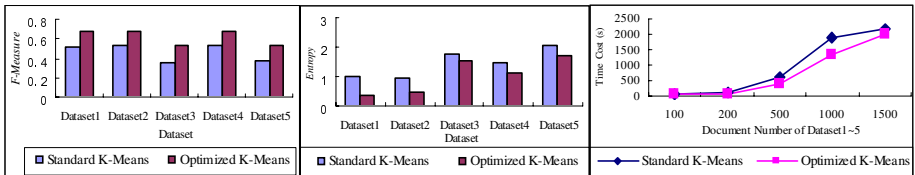


Fig. 1. Comparison about F-Measure

Fig. 2. Comparison about Entropy

Fig. 3. Comparison about Run Time

6 Related Work

In dimension reduction and quality improvement of document clustering, reference[6] proposed an efficient two-level neural network based document clustering architecture. The idea is similar with ours, but we introduce scalar factor from Self-Organizing Map into the process of clustering without special dimension reduction. In reducing intra-dissimilarity of cluster, reference[8] analyzed the reasons of skew generation and proposed a skew prevention method ESPClust for document clustering. Our idea is similar with the idea of the reference[8]. However, the method of reference[8] was based on model for implementing, our method is using scalar factor from Self-Organizing Map.

In the using Self-Organizing Map idea in document clustering, reference[6] mentioned in above paragraph is an example of using SOM to reduce dimensions. Reference[11] described a method of clustering document using a series of 1-dimensional SOM arranged hierarchically to provide an intuitive tree structure representing document clusters. Reference[12] presented an algorithm of based on Tree-Structured Growing Self-Organizing Feature Map for document clustering. These techniques all improved the quality of document clustering to some extent and gave us many good suggestions although different from our method in using SOM.

Moreover, there are many good algorithms in document clustering self [13,10,1]. These techniques including setting k in K-Means, introducing new data structure, identifying important feature, and so on, will provide more improvement ideas for our research.

7 Conclusions

We analyze that the reason of intra-dissimilarity of using K-Means algorithm in document clustering are because of high-dimension and sparseness properties of documents, especially the same influence to means of clusters from all new documents. To solve this problem, we reference to Self-Organizing Map (SOM) and introduce the scalar factor into K-Means, the most popular algorithm of document clustering. The basic idea of K-Means with scalar factor is to modify the means of clusters supervised by scalar factor as the coming of documents in the assignment step. From experiments, this method improves the performance of K-Means in dealing with intra-dissimilarity through calculating *F-Measure* and *Entropy*.

K-Means is a classical clustering algorithm but not the best one, so in our future work, we will compare our algorithm with the related work mentioned in Section 6 to test the quality of our algorithm.

References

1. V.Dobrynin, D.Patterson, and N.Rooney. Contextual Document Clustering. ECIR2004, 2004.4. 167-180
2. X.Hang and H.Dai. An Immune Network Approach for Web Document Clustering. WI2004, 2004.11. 278-284
3. T.Honkela. Description of Kohonen's Self-Organizing Map. <http://www.mlab.uiah.fi/~timo/som/thesis-som.html>, 1998-1-2
4. A.Hotho, S.Staab, G.Stumme. WordNet Improves Text Document Clustering. SIGIR 2003 Semantic Web Workshop, 2003. 10
5. C.Hung, S.Wernter, and P.Smith. Hybrid Neural Document Clustering Using Guided Self-Organizing and WordNet. IEEE Intelligent System, 19(2), 2004. 68-77
6. M.Hussin, M.Kamel and M.Nagi. An Efficient Two-Level SOMART Document Clustering Through Dimensionality Reduction. ICONIP2004, 2004.11. 158-165
7. M.Kantrowitz, B.Mohit, W.Mittal. Stemming and its Effects on TFIDF Ranking. SIGIR2000, 2000.7. 357-359
8. X.Li, G.Yu, D.Wang and Y.Bao. ESPClust: An Effective Skew Prevention Method for Model-Based Document Clustering. CICLing2005. 2005.2. 735-745
9. D.Modha, W.Spangler. Feature Weighting in k -Means Clustering. Machine Learning, 2003, 52(3). 217-237
10. Z.Niu, D.Ji, and C.Tan. Document Clustering Based on Cluster Validation. CIKM2004, 2004.11. 501-506
11. B.Russel, H. Yin, and N.Allinson. Document Clustering Using the 1 + 1 Dimensional Self-Organizing Map. IDEAL2002, 2002.8. 154-160
12. X.Zheng, W.Liu, P.He, and W.Dai. Document Clustering Algorithm Based on Tree-Structured Growing Self-Organizing Feature Map. ISNN(1), 2004.8. 840-845
13. L.Zhuang and H.Dai. A Maximal Frequent Itemset Approach for Web Document Clustering. CIT2004, 2004.11. 970-977

Hierarchical Metadata Driven View Selection for Spatial Query Evaluations

Songmei Yu

MSIS Department and CIMIC,
Rutgers University, NJ, USA
songmei@cimic.rutgers.edu

Abstract. A spatial data warehouse (SDW) is constructed to support the spatial data analysis for decision support purposes. Selectively materializing spatial views to rewrite input queries and thus reduce query response time is a challenging issue for spatial query evaluations. In this paper, we first investigate the issue of using spatial metadata to construct a spatial view dependency framework, which implies an order to materialize views. We then propose a cost model to evaluate the cost for processing spatial queries, which measures the online computation vs. space cost for spatial queries. A greedy algorithm is introduced to materialize a set of views based on the view dependence framework with associated cost value at each dependence level, which shows the local cost optimality of designing an SDW.

1 Introduction

A spatial data warehouse (SDW) is introduced in [1], which is defined as a subject-oriented, integrated, time-variant, and non-volatile collection of both spatial and non-spatial data in support of management's decision-making processes. An SDW consists of a set of materialized views defined over the source relations, either conventional, spatial, or both combined. Some views may also be defined over other views. In order to ensure high query performance, the input queries into an SDW are answered by either complete or partial rewriting over the materialized views.

We consider two important issues regarding selective view materialization. The first one is to evaluate the dependence between views, i.e., how one view can be computed from another. Given a set of input queries, finding an appropriate way to represent the dependency order among them becomes a challenging task. The second issue is the cost value associated with each spatial view. In this paper, we investigate the cost issues for both storage overhead and online processing of spatial queries. Eventually, we develop a greedy algorithm to generate a set of views for materialization to achieve the cost optimization in an SDW, which combines the spatial view dependency and cost considerations at each view dependence level. Our work significantly extends the traditional data warehouses to spatial domains by considering the unique features of spatial

data types and spatial operations, and guarantees the local cost optimality for designing an SDW.

Let us look at a motivation example. We have a set of spatial objects as administrative maps with their alphanumeric counterparts, as well as three basic metadata: *location*, *time*, and *resolution*. We use a multi-dimensional star model to represent the maps and their metadata in figure 1.

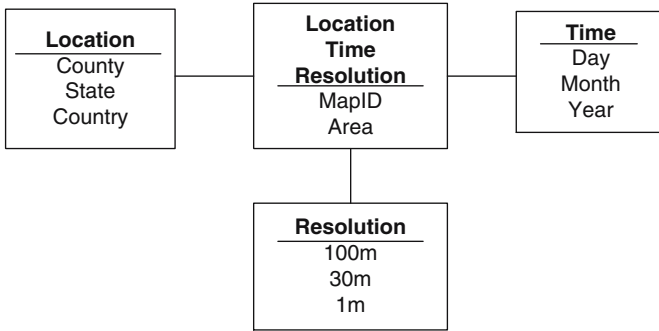


Fig. 1. A simple spatial data warehouse

A concept hierarchy for each dimension can be created by users or experts or generated automatically by the data analysis. For example, we could have a simple total order as a concept hierarchy for each dimension, Day-Month-Year-None for *time*, County-State-Country-None for *location*, and 1000m(low)-30m(mid)-1m(high)-None for *resolution*. The challenging issue is how to build a dependency framework to represent the inter-relationship among the spatial views. For a single dimension, this is simple as long as we have a hierarchical structure for it. However, how to measure the dependencies between spatial views built on several dimension hierarchies, how to quantify the generality of a view, and how to deal with a partial order for a dimension hierarchy, etc, all of these issues demand a powerful data structure and corresponding techniques.

2 The 3D View Dependency Framework

We consider dimensions T (time), L (location) and R (resolution) as three basic metadata for all spatial data warehouses. Assume we have hierarchical positions $(T_1, \dots, T_i, \dots, T_k)$ for dimension T , hierarchical positions $(L_1, \dots, L_j, \dots, L_m)$ for dimension L , and hierarchical positions $(R_1, \dots, R_p, \dots, R_n)$ for dimension R . The positions in a dimension form a total order in specific-general way. Then the 3D dependency framework is formed by connecting three positions from three dimensions respectively, which results in $k \times m \times n$ triangles with each triangle represents a spatial view associated with three dimension attributes. We denote each triangle as $\Delta(T_i L_j R_p)$.

For example, figure 2 (a) and (b) present two sub-parts of the view dependency framework of the motivation example. Three axes represent three dimensions respectively, with each unit of an axis representing a position of the hierarchy on that dimension in an ascending order. For instance, we have $Day(D) - Month(M) - Year(Y)$ for T , $County(C)-State(S)-Country(U)$ for L , and $1m(R_1) - 30m(R_2) - 1000m(R_3)$ for R . A triangle \triangle is formed when we connect three hierarchy positions from three dimensions respectively. There could be $3 \times 3 \times 3 = 27$ triangles generated for three dimensions, given 3 hierarchy positions within each dimension.

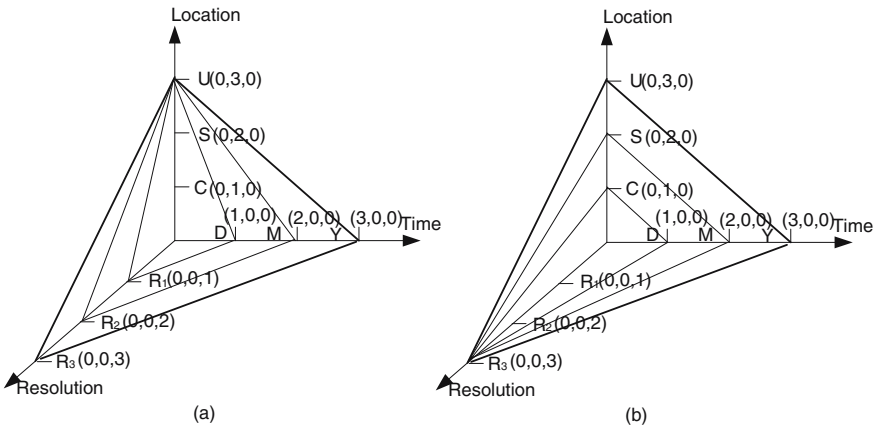


Fig. 2. A sample 3D view dependency framework

We quantify the generality of a view, i.e., the abstract level of a view, in a 3D view dependence framework by computing the perimeter of the represented triangle, which is the sum of *Euclidean distance* of each side within the triangle. We observe that the larger the value of the perimeter for a triangle, the more general the view represented by that triangle. Generally, the view represented by a triangle with larger perimeter value depends on the view represented by a triangle with smaller perimeter value. If two triangles have the same length of perimeter, then the two views have the same generality and there is no dependence relationship between them.

The *direct dependence* relationship between any two views in a 3D view dependence framework, denoted as \prec , is defined as follows. Suppose there are two views V and V' , by denoting $V \prec V'$, we mean that V can be computed by performing certain aggregation functions, either spatial or non-spatial, only on V' , or V completely depends on V' . Otherwise, there is no direct dependence relationship between V and V' . We develop a proposition for identifying the direct dependence relationship between any pair of views in a 3D view dependence framework as follows: $V \prec V'$ if and only if V has one level higher hierarchical position than V' on at least one dimension, and other two dimensions of V have

either equal hierarchical position to or one level higher hierarchical position than the respective dimensions of V_l . The detailed proof is in [2].

Now we discuss the extendibility of a 3D dependency framework when (1) the number of dimensions increases, or (2) the hierarchy within a dimension is a partial order, or a lattice. Consider a set of dimensions D , with each dimension has hierarchy denoted as $D_i(H_j)$, where $i \in (1, n)$, $j \in (1, k)$. The hierarchy on each dimension forms a total order in specific-general way, i.e., a view on H_j is more general than the view on H_{j-1} . An n -dimensional dependency framework is built by connecting each hierarchical position of one dimension to all positions of any other dimensions. Thus a hyper-plane is formed to represent a view. The total number of generated view is, $N = \prod_{i=1}^n K_i$. A triangle in 3D platform is the simplest hyper-plane. Although it is difficult to envision an n -dimensional hyper-plane, we can still use the perimeter of the hyper-plane to quantify the generality of a view, which is computed as the sum of Euclidean distance of each side connecting two hierarchical positions from two dimensions respectively. It is also straightforward to extend the direct dependency relationship into an n -dimensional dependency framework.

If the hierarchy on one or more than one of dimensions is in a partial order, the view dependency lattice will be constructed differently. For example, figure 3(a) describes a simple lattice on *Time* dimension. We can see due to the partial order, we cannot represent a dimension on one axis as we normally do for a total order. We modify figure 2(a) in figure 3 (b), where the *Time* dimension is split into two sub-dimensions *Time1* and *Time2*, and the coordinate system is built accordingly. In figure 3 (b), we refer to the view represented by the triangle $\triangle UR_3W$ as V_1 and the view represented by the triangle $\triangle UR_3Y$ as V_2 . V_1 has larger perimeter value than V_2 , thus V_1 is more general than V_2 . However, V_1 shows no dependence on V_2 because we don't compare the hierarchical position level between Y and W as they are not in the same dimension.

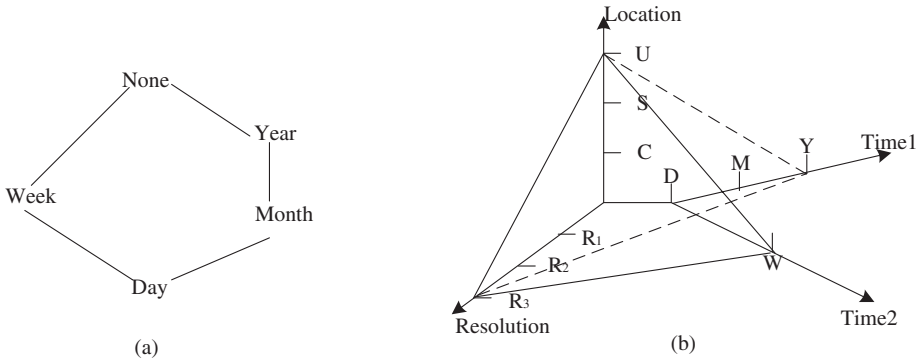


Fig. 3. A dependency with partial order

3 Cost Model

A spatial query typically consists of several atomic spatial operations, which are either spatial selections or spatial joins. If we decompose the query into component queries, the component query with only one single spatial operation is denoted as a simple spatial query (SSQ). We use q to denote an SSQ and p to denote a spatial query composed of several q , and each q is an un-divisible unit within p . The following three factors should be considered to decide whether a q needs to be computed on-the-fly or be materialized: (1) **The on-the-fly computation cost.** We measure the on-the-fly computation cost of a q , denoted as $P(q)$, in terms of the query response time. Most of spatial query processing methods, represented by spatial operators, fall into two categories: spatial join and spatial selection (include point or range queries). Theodoridis et al. [3] have presented analytical models to estimate the response time for both selection and join queries using R-trees. We adopt the unified cost model proposed by them to estimate the cost of $P(q)$. (2) **The potential access frequency.** Based on the access history one may estimate the access frequency of q , which is denoted as $fr(q)$ over a certain time period. (3) **Size.** Basically, an image/map size is measured in *Bytes* or *Megabytes* and determined by several factors, such as resolution (the number of pixels per inch) and channel (Bytes per pixel). We assume the size of the generated view from q , denoted as $S(q)$, is a fixed value at the data warehouse design level, by specifying the height, width, resolution and channels of the generated view. Given the maximum space capacity of the SDW is M , obviously one must materialize a view only if $S(q) \leq M$.

Now we have $P(q)$, $fr(q)$ and $S(q)$, we define the execution cost of q denoted as $C(q) = (P(q) \times fr(q)) / (S(q) / M)$, the total cost of a spatial query $p = P(p_o) + \lambda(p_m)$, and the total cost of an SDW as $C(Q, V) = \sum_{i=1}^n C(p_i)$.

Basically, for each q in an SDW, the more frequently or expensively to compute it or the less space its result takes, the larger the value of $C(q)$ would be, and the more likely we need to materialize it. Therefore, we need a control threshold, which is denoted as δ and used to control the cost of executing a q . The general rule is that if $C(q)$ is greater than δ , we materialize q otherwise we compute it on-the-fly. By controlling the execution cost of each q within a spatial query p , we divide p into two subsets, one is denoted as p_o for materialization and the other is denoted as p_m for on-the-fly computation, and $C(p)$ is the total cost of a given spatial query p where λ is a parameter between 0 and 1 indicating the relative importance of the spatial query computational cost vs. the space cost. Given the set of queries $Q = \{p_1, \dots, p_n\}$ and a set of materialized views V from Q , the total cost of an SDW is computed accordingly.

4 Materialize Views in a Spatial Data Warehouse

Suppose we are given a view dependency framework for an SDW, with the total cost value associated with each view. The basic idea is to take a part of queries for materialization and identify another part of spatial operations for

online computation, by controlling the execution cost value less than the cost threshold. We first define a *benefit function* to monitor the cost savings for an SDW. Let A be an arbitrary set of SSQs. The benefit of A with respect to V , an already selected set of materialized views, denoted as $B(A, V)$, is defined as: (1) If $C(Q, V \cup A) < C(Q, V)$, then $B(A, V) = C(Q, V) - C(Q, V \cup A)$, otherwise $B(A, V) = 0$. (2) $B(A, \phi)$ is called the absolute benefit of A . Specifically, we compute the benefit of A by considering how it can reduce the total cost of a given SDW. For each set of SSQs we choose to materialize, we compare the total cost of the SDW with the one before the transformation. If A helps, i.e., the current cost is less than before, then the difference represents the benefit of selecting A for materialization, otherwise, there is no benefit. Now we define a spatial greedy algorithm (Algorithm 1) to select a set of SSQs to materialize.

We consider this spatial greedy algorithm as a cost driven technique deriving a sub-optimal solution for the selective materialization of spatial views, because it commits to a local maximum cost benefit at each iteration of step 2, although not every locally maximum choice can guarantee the global maximality. This approach is always fairly close to optimal and in some cases can be shown to produce the best possible selection of views to materialize given an SDW.

Algorithm 1. The spatial greedy algorithm for view materialization

Require: $Q, P(q), fr(q), S(q)$ for each $q, M, C(p)$ for each p, δ , and λ .

- 1: $V = \phi$.
 - 2: **for** each query p in Q with $C(p) \geq \delta$ **do**
 - 3: **if** $B(A, V)$ is maximized and $S(V) \leq M$ **then**
 - 4: $V = V \cup A$
 - 5: **end if**
 - 6: **end for**
 - 7: Materialize all views in V
-

5 Conclusions

View materialization is an essential query optimization strategy for decision-support applications. In this paper, we first investigated the problem of constructing a view dependency framework based on the concept hierarchies of metadata for a spatial data warehouse. Secondly, we investigated the cost issues of spatial views, where a cost model is developed for measuring spatial queries, which considers query disk access time, frequency and space. A benefit function and spatial greedy algorithm are finally constructed to filter each simple spatial view within spatial queries for materialization by considering its dependence level with associated cost value. We are currently doing concrete implementations to test the algorithm, which may lead to certain improvements for the efficiency.

References

1. Stefanovic, N., Jan, J., Koperski, K.: Object-based selective materialization for efficient implementation of spatial data cubes. *IEEE Transactions on Knowledge and Data Engineering(TKDE)* **12** (2000) 938–958
2. Yu, S., Atluri, V., Adam, N.: Towards view materialization in a spatial data warehouse. Technical Report (2005)
3. Theodoridis, Y., Stefanakis, E., Sellis, T.: Efficient cost model for spatial queries using r-trees. *IEEE Transactions on Knowledge and Data Engineering(TKDE)* **12** (2000) 19–32

Priority Processing in the Web Service-Workflow Architecture

Hwa-Young Jeong

Faculty of General Education, Kyunghee University,
1, Hoegi-dong, Dongdaemun-gu, Seoul 130-701, Republic of Korea
hyjeong@khu.ac.kr

Abstract. Service Oriented Architecture (SOA) is an architecture that is made up of components and interconnections that stress interoperability and location transparency. A service is a grouping of components and these services are joined together by the processes to form a business process. Web services are software components that are self-containing, self-describing modular applications that can be published, located, and invoked across the Web. It may combine several applications that a user needs. Component communication in web service interacted FIFO order by standard network protocols with RPC method. In this paper, I propose component interaction method in web service by priority service processing. So, it is possible that proposal techniques composite and operate multiplex web service efficiently.

1 Introduction

Service Oriented Architecture(SOA) is an architecture that is made up of components and interconnections that stress interoperability and location transparency. SOA contains components, services and processes. A service is a grouping of components and these services are joined together by the processes to form a business process[4]. The web service architecture is based upon SOA pattern. Web services standards can offer lower cost options for enterprise integration, and have helped to promote the emergence of a new class of integration tool[7]. The essential benefit they bring to application integration is standardization[1]. The following basic specifications originally defined the Web Services space: SOAP, Web Services Description Language(WSDL), and Universal Description, Discovery, and Integration(UDDI). SOAP defines an XML messaging protocol for basic service interoperability[2]. The web service concept represents a next generation architecture that provides high level of interoperability between software components based on software industry standards[4]. Web services are a relatively new way to distribute component architecture across network[8]. Key to Web service design are how a Web service processes requests at the server side (its processing model) and how clients invoke and use the service(the Web service's interaction model). The interaction model encompasses the client's interaction with the Web service. A client's interaction with a Web service can be synchronous or asynchronous, or a combination of both types of communication. When a synchronous model is followed, the application typically uses an RPC-oriented interaction. That is, interaction between web service, include component, are

operated by RPC based FIFO method. But this technique increases the inefficient in processing and operating in case connecting web service increase. Also, RPC based FIFO method interaction can not carry out many components' request in web service processing in unit time on the aspect of whole system operation.

In this research, I designed and implemented the interaction workflow with priority method. I set up the priority order of the request processing by component specification in web service. And the component applied in this proposed method is based on EJB, the server side of component model. So, I improved the operation between components in web service.

2 Related Work

2.1 Java Based Web Service-Workflow

Web services follow the service-oriented architecture model shown in Figure 1. In a Java programming environment, the Web service uses the Java API for XML Registries (JAXR) to publish itself to the registry[3].

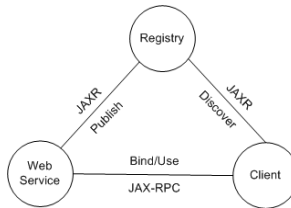


Fig. 1. Publish-Discover-Bind model of web service

The service handles the client request and returns a response at some later point, at which time the client retrieves the response and proceeds with its processing. Enterprise Application Integration(EAI)[10] uses Web services to integrate enterprise information systems. A Web service for EIS integration can be a synchronous or asynchronous service, and this decision is based on the nature of the service's workflow. The architecture of a Web service providing EIS integration might have a JAX-RPC[9] servlet endpoint to receive the client request[3].

2.2 Request and Response Processing in the Web Tier Interface

Web services are a relatively new way to distribute component architecture across network. Components offer a set of services, usable with SOAP protocol[5]. The Web tier generally receives client requests. Incoming client requests (in the form of SOAP messages) are mapped to method calls on the classes that implement the Web service interface. Response generation, which is simply constructing the method call return values, is also performed on the Web tier, again as close as possible to the service endpoint. Keeping this functionality in the endpoint enables you to implement caching of data to avoid extra trips to the EJB tier. The Web service endpoint handles all

incoming SOAP requests and delegates to the business logic exposed in the EJB tier, possibly through a Session Facade design pattern in the EJB tier. Figure 2 shows the recommended way to handle requests and responses in the Web tier interface[3].

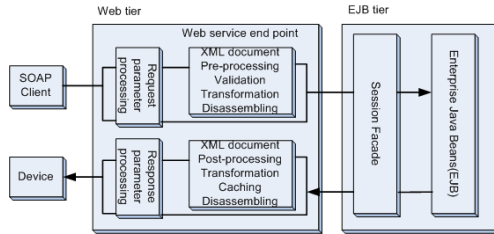


Fig. 2. EAI Architecture with web service. In this model, the endpoint delegates the request to the service's business logic, which in turn accesses the EIS or legacy system through the Connector architecture.

3 Component Interactions in Web Service

3.1 Priority Order of Component Process

In this research, I designed the component interaction method in web service. The component's quality characteristic value is provided in the component specification. Bean request processing time and memory use rate of the component were selected as the applied efficiency characteristic factor. The setting range of the selected efficiency characteristic was assigned in proportion to that value. The efficiency characteristic value of the component was converted into the relevant weight. Therefore, the sum of each weight represents the relative value to the efficiency characteristic value of requesting components. Components which have minor value is handled and performed first. But, in case the sum of weight of requesting components is equal I carried out with FIFO method.

3.2 Priority Process Workflow

A workflow invokes component services by sending messages to front-end modules. For a service expected to give an immediate response, the workflow will call the front-end synchronously(i.e., a WSDL request-response operation). Synchronous messaging with the front-end module is less vulnerable than with the original server since the messaging is done inside server clusters managed by a single service manager. If the front-end does not require communication with the backend during execution of the workflow request, the response time will be kept short. System architecture shows in Figure 3[6].

I modified (A) part in Figure 2 for component(EJB)'s priority process. That is, I insert priority process into Figure 2 within the (A) part in Figure 3. Figure 4 shows the proposed workflow which insert connector for priority process.

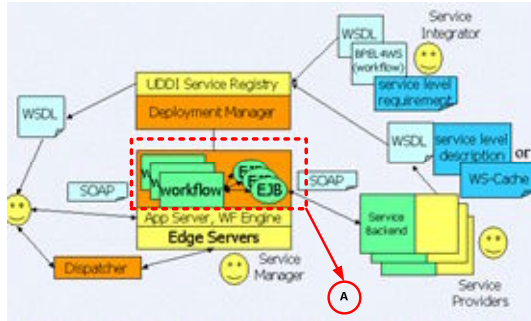


Fig. 3. System architecture

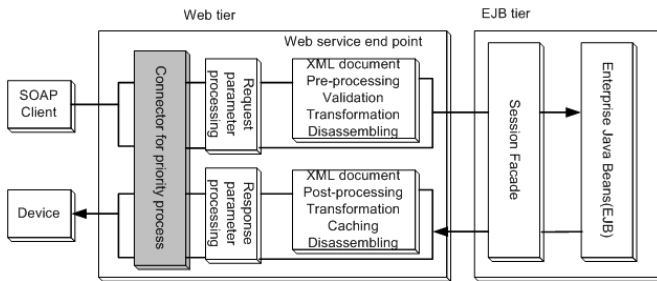


Fig. 4. System architecture

3.3 Design and Implementation of Connector for Priority Process

In this research, applied component model in web service is EJB, server side component model. Figure 5 shows the class and sequence diagram to embody this system.

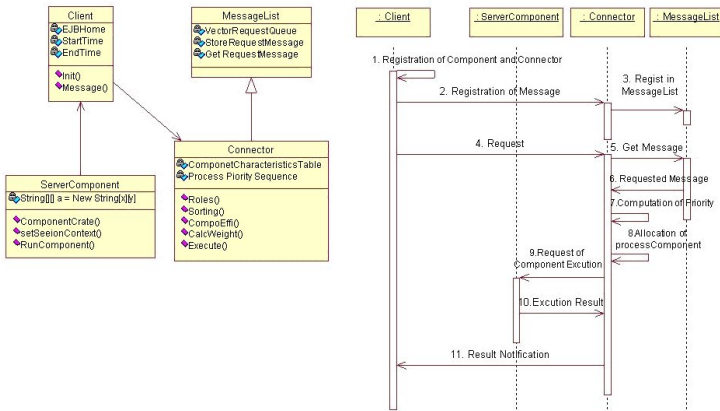


Fig. 5. Class diagram and Sequence diagram

In the class diagram, client shows the request and result from the constructed whole system and registers implemented EJB components through the initialization. In the connector part, I set the attribute table according to the specification of each component and the processing priority sequence based on the processing time and used the bubble sorting algorithm to line up priority order estimations. In the sequence diagram, at the initial step in the client, the composition components were registered and the requesting messages were stored in the message queue.

4 Results Analysis

For operating and testing this system, I implemented 10 example EJB components which have different efficiency characteristics as Table 1.

Table 1. Efficiency characteristics of sample EJB components

Component characteristics	Sample component ID									
	1	2	3	4	5	6	7	8	9	10
Bean request processing time(ms)	690	679	2644	2812	4122	5399	6423	2832	2825	690
Memory Use Rate(%)	6.36	7.01	7.01	7.11	7.11	7.28	7.01	7.01	7.00	6.80

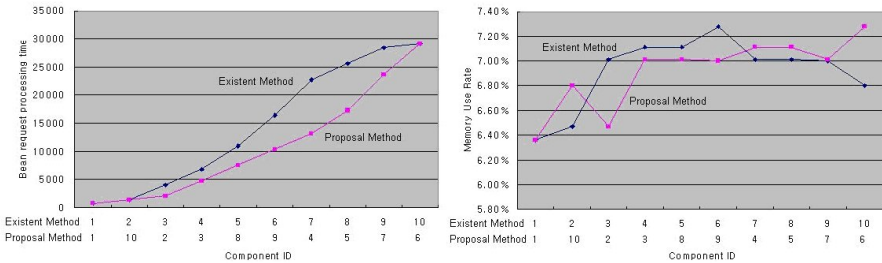


Fig. 6. Bean Request Processing Time and Memory Use Rate

Figure 6 shows test result of the bean request processing time and the memory use rate. In the bean request processing time, on the basis of 15,000ms as the processing time, the existing technique carries out component's ID 1, 2, 3, 4 and 5 but the proposed technique handles component's ID 1, 2, 3, 5, 8, 9 and 10. In the memory use rate, the proposed technique decides the processing order considering each characteristic at a time and handles first the component with the small memory use rate as a whole.

5 Conclusions

This research makes connector's priority process by component specification in web service. Component specification assigned considering the bean request processing time and memory use rate. That is, the assignment in the connector does not depend

on the requested order of the component but the request of the small unit component with the small memory load and the fast bean request processing time should be carried out preferentially. As these results, the efficient processing assignment and operation of whole composed components is possible.

But the proposed technique is demanded more precise structure definition that can not only play a general role in processing the components in the connector but also reflect the references of components' specifications to the request processing.

References

1. Boualem Benatallah, Fabio Casati, Farouk Toumani, and Rachid Hamadi.: Conceptual Modeling of Web Service Conversations. LNCS 2681 (2003) 449–467
2. Tony Andrews., et al.: Business Process Execution Language for Web Services Version 1.1. BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems (2005)
3. Using Web Services Effectively. <http://java.sun.com/blueprints/webservices/using/web-servbp.html>. Sun Microsystems (2002)
4. Anish Joshi.: Active Web Alert Service for Rule Based Alerting in Sensor Web An Event Based Approach. International Institute for Geo-information Science and Earth Observation MS Thesis (2005)
5. Jarmo Korhonen, Lasse Pajunen, and Juha Puustjärvi.: Using Web Services and Workflow Ontology in Multi-Agent Systems. EKAW'02 workshop on Ontologies for Multi-Agent Systems (2002)
6. Junichi Tatemura, Wang-Pin Hsiung, Wen-Syan Li.: Acceleration of Web Service Workflow Execution through Edge Computing. The Twelfth International World Wide Web Conference (2003) 20-24 May
7. Nigel Thomas, Warren Buckley.: The enterprise service bus: a developer-friendly integration engine. Web Services Journal (2003) Oct
8. J. Korhonen, L. Pajunen, and J. Puustijarvi.: Using Web Services and Workflow Ontology in Multi-Agent Systems. EKAW'02 workshop on Ontologies for Multi-Agent Systems (2002)
9. Java API for XML-Based RPC 1.1 Specification. <http://java.sun.com/xml/jaxrpc/index.html>. Sun Microsystems
10. John Godel.: Web Services Architecture. <http://www.devnewz.com/devnewz-3-20030708/WebservicesArchitecture.html>, iEntry, Inc. (2003)

Policy-Based Workflow Management System

Shi Chen, Song Ouyang, and G.K. Hassana

Department of Computer Science, Central South University, Changsha, Hunan,
P. R. China 410083
Tel. +86 731 8876887
christiancskkk@163.com, ouyangsong@yahoo.com

Abstract. One of the most serious limitations of traditional workflow management systems is lack of flexibility. This paper considers one of the main factors causing poor flexibility is the imperative nature of process management, which forms the foundation of traditional workflow system. To combine the advantages from both the policy management and the process management, a prototype policy based workflow management system (PWfMS) is presented. A meta-model for PWfMS is developed to support the system design and implementation. An extension to Petri net, Variation Petri Net system (VPN), is proposed to support the modeling and the analysis for new approach.

1 Introduction

Workflow management technologies have been used in more and more organizations and applications to automate, manage, and improve the business processes. However some problems are still not solved in the traditional workflow management system (WfMS). One of the most serious problems is lack of flexibility [1]. To improve the flexibility, Hu et al. propose a three-layer workflow concept framework to realize enactment flexibility [2]; Ellis et al. use a Petri net formalism to analyze structural changes [3]; Reichert et al. present a formal foundation to support the dynamic structural changes of running workflow instances [4]; in [5] Weske defines a formal foundation and conceptual design of dynamic adaptations in an object-oriented workflow management system and describes how workflow schemas are represented. Although considerable work on improving the flexibility has been done, the results are not satisfactory enough and there is no agreement on the best solution of this issue.

This paper considers that one of the main factors causing poor flexibility problems is the imperative nature of process management, which forms the foundation of traditional workflow systems. Some problems are: the system operators need know too many information about the processes to be executed before the designing new workflow schemas; the process models are often over-specified and are difficult to implement; the process expression and the process implementation are usually in same component; the build-time functions are bound with the run-time functions too tightly, etc. These result in many flexibility problems. This paper presents a policy based workflow management system (PWfMS). The idea behind the PWfMS is to combine the policy management and process management and take the advantages from both.

2 A Metamodel for New Approach

To describe the workflow management system more accurately, a process definition meta-model is proposed by WfMC [6]. A meta-model can be used to express the objects, their relationships and attributes within a process definition. Based on the meta-model proposed by WfMC and the concept of integration of policy management and process management, a meta-model for new approach is developed to support the design and the implementation of new system. This model is shown in Fig.1 below.

Comparing with the WfMC’s meta-model, the main differences between the two models are: (1) the “Activity” in the WfMC’s model is changed to “Abstract Policy”; (2) a Policy Repository is added. The abstract policy becomes the basic element in the workflow definition. An abstract policy consists of “PE:Policy Event”, “PC:Policy Condition”, and “PA:Policy Action”. The Policy Repository contains basic policy model data for specific domain.

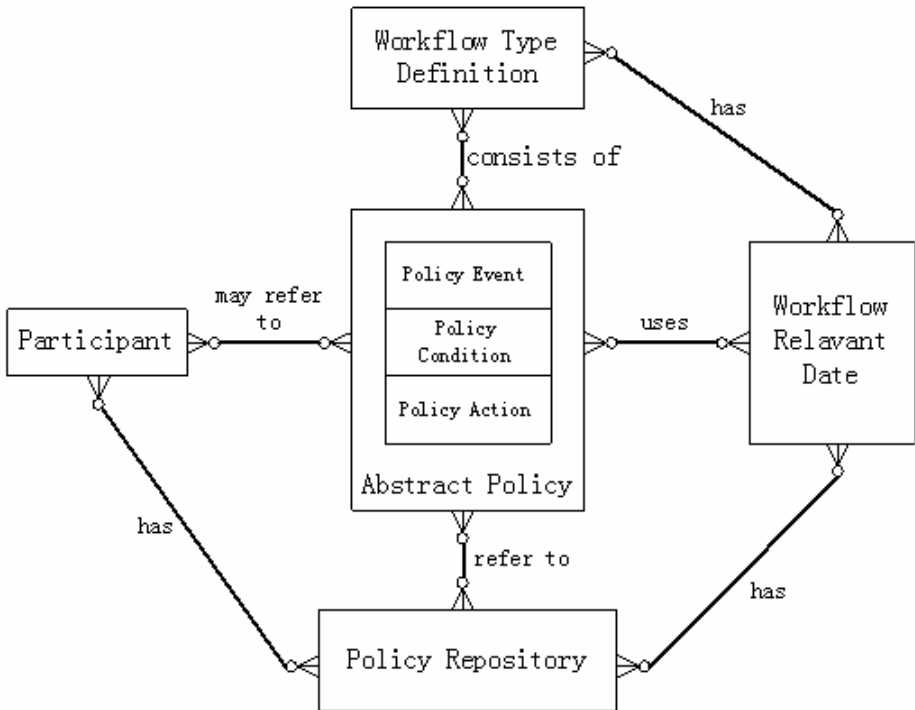


Fig. 1. Policy Based Process Meta-Model

Policy management is defined as the usage of policy rules to manage the behavior of one or more entities. This methodology describes how one or more applications can manage one or more systems according to a set of policy rules. The proper combination of policy management and process management can bring much better flexibility to workflow management systems.

To implement the new approach, two basic types of policies are needed: (1) policies to represent business rules, and (2) policies to represent the commands for executing the processes. They are referred as abstract policy and concrete policy respectively. In general, a policy model can have many levels; only the lowest level policies representing the commands directly executed by the supporting platform. All higher-level policies can be translated into lower level policies in run time. The translation work is carried out by policy-server. The multilevel architecture of policy model can help better integration of policy management and process management

3 Extension of Petri Net for New Approach

To use Petri net for the new approach, some extensions have to be made to deal with the specific characteristics of the new approach.

3.1 Variation Petri Net System

The extended Petri net is referred as Variation Petri Net System (VPNS) in this paper. The extensions are described below:

3.1.1 Node Extension

There are two purposes for node extension: (1) To distinguish OR-split and And-split in the graphic level. (2) To model the Election Routing that is specific in policy management. In the new policy-based system, the split can only be initiated by transitions, therefore three special transitions are introduced in VPN system; they are AND-split, OR-split and Election-split as shown in Fig.2.

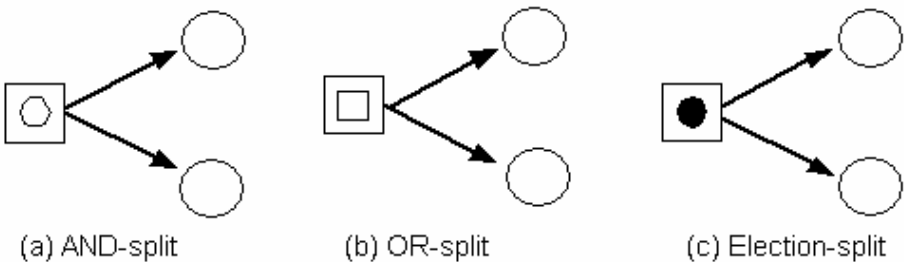


Fig. 2. Three types of split in VPNS

3.1.2 Weight Function Extension

Weight function of Petri net system is a constant ($\equiv 1$), that means every arc can fire only one token of its predecessor place node. This is not suitable for the system introduced in this paper. Weight function (W) of VPN system has following properties:

- The output arc of each transition has a weight that stands for the weight of each split path corresponding to a policy action.
- Every input arc will produce tokens of the same number as the weight in its output place.

- Places have weight too. Only when the number of tokens that a place holds is equal to or greater than its weight, can the successor transition of this place become enabled.
- Weight of the output arc of a place can only take two values: zero or infinity. In the case of zero weight it means that this arc will fire all the tokens of its predecessor place, but won't alter the weight of this place. In the case of infinity it means that this arc will fire all the tokens of its predecessor place and change the weight of this place to infinity in the same time.

In VPN system, all the split paths can only join to places, and the join rule is based on the weight function. The three types of join nodes, of which each type corresponds to a split node type, are shown in Fig 3.

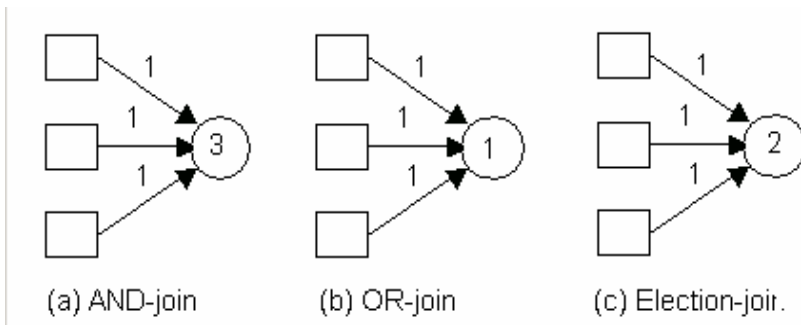


Fig. 3. Three types of join in VPNS

3.2 Model Analysis

The analysis of soundness property of Petri net can be used to avoid the fault errors. In the following subsections a method is developed to analyze the VPN system. To make the analysis easier, two modified structural characterizations are introduced below:

3.2.1 Free-Choice

A free-choice Petri net means that the routing of a process should be independent of the order in which activities are executed.

Definition 3.1 (Free-choice) [7]: A Petri net is a free-choice Petri net if for every two transitions. T_1 and T_2 , $\bullet T_1 \cap \bullet T_2 \neq \emptyset$, implies $\bullet T_1 = \bullet T_2$. In VPN system, a split can only be initiated by transitions and split path can only join to places, which means every transition has only one predecessor place, this structural character guarantees the free choice property of VPN system. That means that if the model of a workflow is built based on VPN system, it then has free-choice property.

3.2.2 Well-Handled

The right association between AND/OR-split nodes and AND/OR-join nodes is required for a soundness workflow. The split route paths initiated by an AND

(OR)-split node should not join to an OR(AND)-join node, which is well-handled property of Petri net.

Definition 3.2 Well-handled VPN: A VPN is well-handled, if for any pair of nodes x and y such that x is a transition, y is a place, $f(y)$ is the weight of input arc of y , $W(y)$ is the weight of y itself, and for any elementary paths: $C_1, C_2, \dots, C_n \rightarrow a(C_i) \cap a(C_j) = \{x, y\}$ ($1 \leq i, j \leq n, i \neq j$), then the following three conditions must be met: (1) If x is an AND-split transition, then $W(y) = n * f(y)$; (2) If x is an OR-split transition, then $W(y) = f(y)$; (3) If x is an ELECTION-split transition, then $f(y) \leq W(y) \leq n * f(y)$.

3.2.3 Election Routing Model

Election routing is a special routing one can find in VPN. In fact, its split transition type is AND-split, but its join node is a special one. In election routing, if the number of concurrent rout paths is n , those paths would not join to a node until a certain number of paths complete execution. Just like the case of an election, the result can not be reached until certain number of the decisions have been made. This certain number is defined in the build-time of workflow process, suppose it to be j , and for the ELECTION-join node y : $W(y) = j * f(y)$ ($j \leq n$). Fig.4 shows the election routing of VPN system.

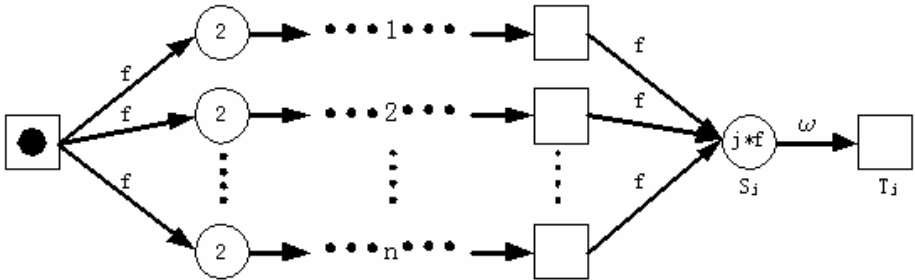


Fig. 4. A Election routing topology

Election routing will convert to parallel routing when $j=n$, which is called **full-election**. If it is not well-conditioned, an election routing will lead to reduplicated execution of same transitions. This error is described in detail as follows:

In Fig.4, considering $j = \lceil n/2 \rceil^1$, when j of n paths are completed, then S_j is enabled; which leads to the execution of T_j . Since all the paths are concurrent, the remained paths are continue running, the number is $j' = n - j$, and $j = \lceil n/2 \rceil \rightarrow j' > n - (n/2) > j$, if the weights of those remainders, $f * f$, greater than the weight of S_j , $W(S_j) = f * f$, which will enable S_j one more time and consequently, T_j will be executed again. For such a reason, under the condition of $\lceil n/(x+1) \rceil \leq j \leq \lceil n/x \rceil$ ($1 \leq x < \lceil n/2 \rceil$), T_j will be executed for x times.

¹ Here $\lceil x \rceil$ is the integer part of x .

This error is referred as “Minority Error” of VPN in this paper. The solution of it is to modify this rule: since S_j has been enabled once, it should not be enabled again for its weight has changed to $W(S_j) = \omega$ after the execution of T_j as shown in Fig.4.

4 Conclusions

A prototype policy-based workflow system that combines policy management and process management has been developed, and valued experience has been obtained. However, it must be noted that to implement a production system that integrates the policy management and process management there are still many things to be sorted out. Firstly, the suitable policy model, a multi-level structure, is domain dependent. To make the modeling process standard for different domains, a modeling language is expected. Secondly, the policy server, it is the most difficult part of the implementation of the new approach, needs lot efforts to make it working in real applications. Finally, the integration of the policy server and the workflow engine expects more creationary work.

References

1. Aalst, W.M.P. van der, Weske, M.H., & Wirtz, G.: Advanced topics in workflow management. Issues, requirements and solutions. *Journal of Integrated Design and Process Science* 7 (3), 47-77. (2003)
2. Jinmin Hu, Paul W. P. J. Grefen: Conceptual framework and architecture for service mediating workflow management, Vol.45 (2004) pp929 – 939
3. CA Ellis, K. Keddara, G. Rozenberg: Dynamic change within workflow systems, in: Proceedings of International. ACM Conference COOCS '95, Milpitas, CA, (August 1995), pp. 10–21.
4. Reichert, M., Dadam, P.: Supporting Dynamic Changes of Workflows Without Losing Control. *Journal of Intelligent Information Systems, Special Issue on Workflow and Process Management*, Vol. 10, No. 2(1998)
5. Weske, M.: Formal Foundation and Conceptual Design of Dynamic Adaptations in a Workflow Management System, 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 7 (January 03 - 06, 2001)
6. Workflow Management Coalition (2002): The Workflow Reference Model. <http://www.wfmc.org/standards/standards.htm>.
7. W.M.P. van der Aalst: Workflow Verification: Finding Control-Flow Errors Using Petri-Net-Based Techniques. In *Business Process Management*, pages 161-183, (2000).

A Mobile-Aware System for Website Personalization

S. Greco, A. Scicchitano, A. Tagarelli, and E. Zumpano

DEIS, University of Calabria, 87030 Rende, Italy

{greco, ascicchitano, tagarelli, zumpano}@deis.unical.it

Abstract. The increasing availability of portable computing devices has enabled wireless networking to support mobile users in experiencing emerging applications and services. In this paper we extend a previously proposed work on supporting users for personalized navigation of websites, to make it *mobile-aware*. The objective is to provide personalization, in form of page recommendations, w.r.t. a specific context of use. In particular, we exploit information on user physical location in detecting navigational preferences, and device/network information to address application adaptability.

1 Introduction

An important research challenge in the Web age consists in enhancing the capability of a Web-based information system to tailor itself to the specific preferences and requirements of users. Web personalization can be effectively accomplished by suitably handling knowledge discovered from the analysis of the dynamic nature of user interactions together with the specifics of the website [3].

Traditional adaptive and personalization systems are typically designed and configured without taking into account the user location. However, the use of portable computing devices capable of wireless networking, also known as *mobile computing*, requires necessarily to consider different factors addressing the context of use, i.e. the user, the information content, the mobile device, and the network.

From a mobile context perspective, traditional personalization issues become more complex, while new ones arise. Indeed, besides website structure and content and user profiles, additional wireless-aware information might be considered, such as user mobility, device features (e.g. interface capabilities, processing power, type of connectivity) and network features (e.g. bandwidth, QoS level, localization capabilities, security). Moreover, the needs of a mobile user should also take his/her location into account, i.e. content delivery should be filtered by the user's geographical position.

Research effort on web personalization for mobile users is still in a preliminary stage, if compared with the results obtained by works on traditional web personalization [3,5]. A precise statement for the website personalization problem is given in [1], although information on device, network and user location is marginally, or not at all, considered. A mobile agent based architecture, which combines existing techniques in a component-based fashion, is proposed in [6]. In [2] the focus is on the design of a decision system for processing and managing the information delivery to the user. The set of parameters chosen to provide adaptive mobile services is mostly complete, and includes device, network and user position. However, decisions are based on simple policies of parameter setting, whereas no personalization action is defined.

Contribution. In this paper we extend a previously proposed work on supporting users for personalized navigation of websites [4]. That work presents an approach that provides a user with a non-invasive, personalized view of any page visited in the website currently accessed. This is accomplished by mainly exploiting knowledge discovered both from user preferences and usage and content correlations among pages. A clustering process guides the identification of users with similar navigational preferences, according to their browsing histories. For any *active user*, the membership cluster is selected to extend the set of pages of interest to those of the users in the same cluster. Usage and content correlations are discovered from these pages, in order to provide a ranked list of interesting suggestions for the active user.

The above approach can be easily enhanced to make it *mobile-aware*, so that an implemented system can also be utilized through mobile devices. Our concern is to provide personalization w.r.t. any *active context of use*, which is represented by the active user, at a particular location, while visiting a particular site page by means of a mobile device through wireless communication. The underlying idea lies substantially on two reasons. Firstly, user location information may act as a discriminatory factor in detecting the user navigational interests. Indeed, pages containing information on certain location-sensitive topics (e.g. restaurants, hotels, etc.) should be relevant specifically to users whose geographical position is close to the target location. Therefore, interests for any active user should be discovered accordingly to a particular location, and in a collaborative filtering fashion, that is by exploiting the profiles of a neighborhood of users whose interests and location are similar to those of the active user. Secondly, device and network information might be exploited to serve for application adaptability. This is strictly related to the design of the personalization action, which in our setting has the form of a menu bar placed on top of the page and organized by sections containing page recommendations.

2 Web Page Recommendation for Mobile Users

In this section we give the statement for the problem of web page recommendation for users of mobile devices. Given a website W , we assume the following information is available:

- A set \mathcal{P} of pages in W . Any page p is represented by its URL and is coupled with a type denoting the p 's function (e.g. home page, index page, content page).
- A set \mathcal{U} of users of W . Any user u in \mathcal{U} is either a past or a current visitor of pages in \mathcal{P} , and possibly has associated a static profile containing demographic information (e.g. age, gender, country) and other potentially useful information on the user's web experience (e.g. favorite channels).
- A set \mathcal{L} of strings denoting geographical locations (e.g. province, city, street names). Any location $l \in \mathcal{L}$ can be computed by converting coordinates received by a GPS device into a geographical location.
- A set \mathcal{S} of user sessions. Any session $s \in \mathcal{S}$ is a tuple $\langle u_i, l_j, P, t \rangle$, where $u_i \in \mathcal{U}$ is the associated user, P is a sequence of pages visited by u_i , $l_j \in \mathcal{L}$ denotes where u_i was while visiting P , and t is a timestamp indicating the end time of s . Moreover, we denote with \mathcal{S}_u , \mathcal{S}_l , and \mathcal{S}_p the historical set of sessions respectively associated to any user u , any location l , and any page p .

- An *active context* (u_a, l_a, p_a) , where u_a , l_a and p_a denote the active user, the active location (i.e. the current location of u_a at time of visiting p_a), and the active page (i.e. the page currently visited by u_a), respectively. The u_a 's session that includes the visit of p_a is associated with a timestamp t_c , which is clearly assumed to be greater than the timestamp of any previous sessions.
- A set \mathcal{D} of mobile devices. Today typical categories of mobile devices include Palms, Pocket PCs, SmartPhones, Tablet PCs, which differ mainly w.r.t. memory capacity, processor frequency, computation power and screen size.
- A set \mathcal{N} of wireless networks, such as WPANs (e.g. Bluetooth), WLANs (e.g. WiFi), and cellular networks (e.g. GSM, GPRS, UMTS).

Let us consider a page p , a user u , and a location l . We select a set of pages related to p and a set of users related to u , w.r.t. location l . More precisely, starting from p , we detect the set $\mathcal{U}_{p/l}$ of users who have visited p when located at l , and then the set $\mathcal{P}_{p/l}$ of pages visited by users in $\mathcal{U}_{p/l}$. Dually, starting from u , we select the set \mathcal{P}_u of pages visited by u , and then the set $\mathcal{U}_{u/l}$ of users who have visited pages in \mathcal{P}_u when located at l . Note that $\mathcal{U}_{p/l}$ can be recognized by the set $\mathcal{S}_p \cap \mathcal{S}_l$ of sessions involving p 's users navigating from location l ; analogously, $\mathcal{U}_{u/l}$ can be recognized by the set $\mathcal{S}_u \cap \mathcal{S}_l$ of sessions involving u 's navigations at location l .

Our approach to web page recommendation for mobile users can be devised in four main stages: *i)* location-based detection of user preferences, *ii)* identification of users with similar preferences, *iii)* discovery of usage and content correlations between pages, and *iv)* computation and ranking of interesting recommendations for the active user. The ultimate objective is to provide any user of a target website with a personalized access that fulfills his/her navigational preferences in a particular context of use.

Location-Based Detection of User Preferences. The preference of a user for a given page can be assessed by suitably weighting the visit frequency of the user during the relative history of accesses to the site. However, for a mobile user, preferences should be measured contextually to a given geographical location. Let $u \in \mathcal{U}$ be a user, $l \in \mathcal{L}$ be a location, and $p \in \mathcal{P}$ be a page. The degree of preference of u for p , when u 's position is l , is provided by the *preference function*:

$$pref(u, l, p)|_{t_c} = \sum_{s_i \in \mathcal{S}_u \cap \mathcal{S}_l} \log_{d_i} (freq(u, l, p)|_{t_i} + 1),$$

where $freq(u, l, p)|_{t_i}$ counts the accesses to p by u , at location l and during session s_i , and $d_i = t_c - t_i$ is the timestamp difference between the current session and s_i .

Given the set \mathcal{U} of users, the set \mathcal{L} of locations, and the set \mathcal{P} of pages, we denote with \mathcal{UL} the set of pairs $\{u/l \mid u \in \mathcal{U} \wedge l \in \mathcal{L}\}$ and with \mathcal{ULP} the $(|\mathcal{UL}| \times |\mathcal{P}|)$ *preference matrix* storing the normalized degrees of preference of all pairs user/location $u_i/l_j \in \mathcal{UL}$ for all pages $p_k \in \mathcal{P}$:

$$\widehat{pref}(u_i, l_j, p_k) = \frac{pref(u_i, l_j, p_k)}{\max_{q \in \mathcal{P}} \{pref(u_i, l_j, q)\}}.$$

Identification of Users with Similar Preferences. The second stage consists in identifying users showing similar navigational preferences. We accomplish this objective by performing a clustering task that accepts as input the following data: the active user u_a ,

at location l_a , with the set \mathcal{U}_{u_a/l_a} of related users, and the active page p_a with the set \mathcal{P}_{p_a/l_a} of related pages. As we have discussed previously, the sets \mathcal{U}_{u_a/l_a} and \mathcal{P}_{p_a/l_a} are selected w.r.t. the active context, i.e. the triple $\langle u_a, l_a, p_a \rangle$. Starting from these sets, a matrix called *active preference matrix* \mathcal{ULP}_a is built to store the normalized degrees of preference of all users in \mathcal{U}_{u_a/l_a} for all pages in \mathcal{P}_{p_a/l_a} .

The output of the clustering process is a partition of the set of users related to the active user in preference clusters, where the notion of homogeneity among users is referred to the preference values stored in the active preference matrix. We consider the set of pages visited by the users assigned to the same cluster of the active user, the *active preference cluster* C_a . We denote with \mathcal{P}_{C_a} this set of pages. The next step is to discover among pages in \mathcal{P}_{C_a} those correlations that lie behind usage and content patterns. We refer to [4] for the adopted clustering technique and the measures exploited to discover usage and content similarities from the set \mathcal{P}_{C_a} .

Computing and Ranking Page Recommendations. We define the *page recommendation function* (PR) as a combination of the user preference function and the page similarity measures. Given the active user u_a at location l_a and the active page p_a , recommending a page $p \in \mathcal{P}_{C_a}$ to u_a w.r.t. l_a and p_a is assessed by the *page recommendation function*:

$$PR_{u_a/l_a}(p, p_a) = \alpha \times f(p, u_a, l_a, p_a) + (1 - \alpha) \times g(p, p_a),$$

where $\alpha \in [0..1]$ weights the user preferences w.r.t. page similarities.

Moreover, $f(p, u_a, l_a, p_a) = \beta \times \widehat{pref}(u_a, l_a, p_a) + (1 - \beta) \times \widehat{pref}(u_a, p_a)$, where $\beta \in [0..1]$ measures the preference of user u_a for the page p when the current location is l_a and the current page is p_a , and $g(p, p_a) = \delta \times sim_U(p, p_a) + (1 - \delta) \times sim_C(p, p_a)$ measures the similarity of the two pages p and p_a , where $\delta \in [0..1]$ weights usage similarity (sim_U) w.r.t. content similarity (sim_C).

We underestimate the recommendation of pages already visited by the active user during the current session, which do not provide any benefit in terms of navigation support. The recommendation of a page p for the active user u_a at location l_a , w.r.t. the active page p_a , is ranked by the *page recommendation ranking function*:

$$PRR_{u_a/l_a}(p, p_a) = PR_{u_a}(p, p_a) \times \left(1 - \frac{freq(u_a, p)|_{t_c}}{n_{u_a}|_{t_c}}\right),$$

where $freq(u_a, p)|_{t_c}$ denotes the number of times page p has been visited in the current session by u_a , and $n_{u_a}|_{t_c}$ is the total number of accesses of u_a in the current session.

3 A System Prototype for Mobile-Aware Website Personalization

The original contents of any visited page should be visualized in understandable forms so that the user is able to easily browse, and the delivery of recommendations should be non-invasive w.r.t. the preexisting structure of the page. For this purpose, we propose a *navigational menu bar* as a light and intuitive personalization interface.

The navigational bar is conceived to be placed on top of any visited page, as an additional layer, and is designed to contain links to pages to be recommended according to the page recommendation function. More precisely, the navigational bar is designed

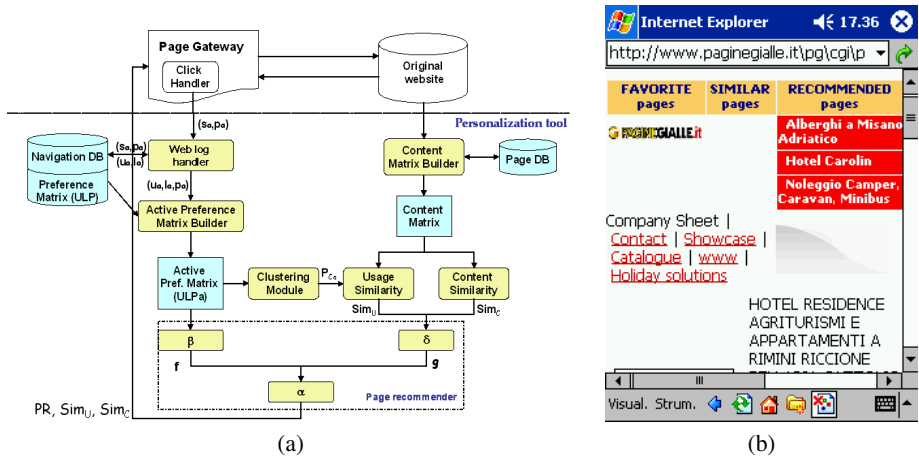


Fig. 1. (a) Architecture of the personalization engine. (b) Personalization of a sample *PagineGialle* page.

with *BS* (*bar size*) sections, or menus, each having maximum length equal to *ML*. Both parameters *BS* and *ML* can be estimated as functions of device and network features. For this purpose, we devise a simple model based on a decision tree, where each internal node denotes a test on a feature, each branch represents an outcome of the test, and leaf nodes represent value settings for *BS* and *ML*. For instance: *if screen* \leq “240×320” and *scrolling* = “yes” and *memory* \geq “32 MB” then *BS* := “3” and *ML* := “5”.

The role of each menu in the bar is to provide a different view of the set \mathcal{P}_{C_a} (C_a is the preference cluster of the active user). The mandatory menu is obviously the one containing links to the ranked *recommended pages*. Other two important menus are *favorite pages* and *similar pages*, that is the menus containing links to pages ranked, respectively, by decreasing degree of user preference and page usage/content similarity. Possibly, the remaining $BS - 3$ are devoted to the recommended pages that have the highest degree of recommendation: for each of these pages, say *p*, the corresponding menu lists the first *ML* pages in \mathcal{P}_{C_a} linked by *p*, ordered by degree of recommendation.

The conceptual architecture of the personalization system is described in [4]. The main module is the *personalization engine* (Fig. 1(a)), which analyzes information about the user behavior from and yields the page recommendations used to build the navigational bar.

3.1 Website Personalization: An Application Scenario

We now present an application scenario that allows us to preliminarily shed light on the behavior of the system when tested against a complex website. For complex website, we intend here a website containing several hundreds of pages, with a complex structure of links among pages, and exhibiting several presentation and content features. Viewed in this respect, we chose the site *PagineGialle* (www.paginegialle.it), an Italian web directory containing information on national companies providing services or products, such as tourism, advertising and services for business, public authorities and communities.

As a test experiment, we considered the following scenario. The active user is located near Rimini, a nice residential holiday centre in the Eastern coast of Italy. The user is mainly interested in accommodation possibilities, but he/she would also like to know any information about travel services. Fig. 1(b) shows a snapshot of a page excerpt from *PagineGialle*. The page was displayed on a web browser running on Compaq Pocket PC iPAQ H3800, and was adapted with a navigational bar having three sections (i.e. BS was set to 3), each with a length (ML) equal to 3. Parameters α , β and δ were set to 0.5.

The recommendations reported on the navigational bar reflected an expected behavior of the system. Indeed, the favorite pages section was related to a cluster of users interested in travel and tourist services not only in Rimini but also in its province (e.g. Riccione). The similar pages section pulled together pages similar by content, such as pages concerning several choices of accommodation in Rimini. Indeed, the expanded menu of recommended pages picked items from the first two menus: in particular, the recommendations “Alberghi a Misano Adriatico” and “Hotel Carolin” refer to hotels in the close neighborhood of Rimini, whereas “Noleggio Camper, Caravan, Minibus” points to a page containing information about car rentals in Rimini.

4 Conclusion and Further Work

We have proposed a mobile-aware approach to website personalization, which aims at providing any user with a personalized access according to a specific context of use, namely the user preferences, the user physical location, information on page content and usage, the device and network features. This approach has been effectively used to develop a lightweight system for computing page recommendations w.r.t. an active context of use. The system has been tested on some significant websites: in this paper, we have presented an interesting application scenario providing encouraging results.

The effectiveness of the system needs to be carefully assessed by carrying out several tests on large websites. In particular, we shall deeply investigate the behavior of the system for different contexts of use, and the role of location in enhancing the user preferences. Finally, a special study will be devoted to find out how much application adaptability can be affected by the mobile device, the network, and the page functions.

References

1. C. Anderson, P. Domingos, and D. Weld, ‘Personalizing Web Sites for Mobile Users’. In: *Proc. of the WWW’01 Conf.*, 565–575, 2001.
2. I. Chisalita and N. Shahmehri, ‘An Adaptive Approach in Providing E-services for Mobile Devices’. In: *Proc. of the CCSSE’01 Conf.*, 13–15, 2001.
3. M. Eirinaki and M. Vazirgiannis, ‘Web Mining for Web Personalization’. *ACM Transactions on Internet Technology*, 3(1):1–27, 2003.
4. S. Flesca, S. Greco, A. Tagarelli, and E. Zumpano, ‘Non-Invasive Support for Personalized Navigation of Websites’. In: *Proc. of the IDEAS’04 Conf.*, 183–192, 2004.
5. B. Mobasher, R. Cooley, and J. Srivastava, ‘Automatic personalization based on web usage mining’. *Communications of the ACM: Personalization*, 43(8):142–151, 2000.
6. C. Panayiotou, G. Samaras, J. Hong, S. Long, R. Kooper, and M. Pinkerton, ‘mPERSONA: personalized portals for the wireless user: An agent approach’. *Mobile Networks and Applications*, 9(6):663–677, 2004.

An Algorithm for Best Selection in Semantic Composition of Web Service*

Juntao Cui¹, Yiwen Zhu¹, Ning Gu¹, Yuwei Zong²,
Zhigang Ding², Shaohua Zhang², and Quan Zhang²

¹ Dep. of Computing and Information Technology, Fudan University,
Shanghai, 200433, China

{cuijuntao, ywzhu, ninggu}@fudan.edu.cn

² Shanghai Development Center of Computer Software Technology, Shanghai, China

Abstract. The automation of Web services interoperability can bring advantages to effective B2B collaboration. However, the techniques for Web service composition may still require a lot of manual efforts. For example, how to select the most appropriate service for service composition is far from trivial. This paper proposes a method for best selection which can perform the most appropriate selection in composition process. Given a selection policy such as minimum cost, the method can find the most appropriate service. Experiments show that the method proposed sometimes only takes a little more time than that without selection policy to select the best composition plan.

1 Introduction

Web services are modularized applications on the Internet which allow applications to communicate with each other [1]. The automation of Web services interoperability can bring considerable advantages to effective B2B collaboration. However, the state-of-the-art techniques for Web service composition still require a lot of manual efforts.

To support service composition, some composition systems were developed such as SELF-SERV [2]. SELF-SERV is a framework through which web services can be composed in different service communities and the resulted composite web services can be executed in a decentralized dynamic environment. However, it emphasizes service execution instead of service composition. Additionally, it employs selection policy such as cost. But it doesn't illustrate the algorithm for best selection in details.

In the previous project, we have developed a composition system termed SASO to enable users to easily compose Web service. In this project, an algorithm without selection policy has been developed to perform service composition, which will choose arbitrary services when there are several services for one specific goal.

The paper presents a method for best selection, which can perform the appropriate selection in composition process, to demonstrate how we can overcome the above-

* This work is supported by the Natural Science Foundation of China(No. 60473124), National High Technology Development 863 Program of China (No. 2004AA1Z2390), Shanghai Science and Technology Committee Key Project for Embedded Systems (No. 04DZ15009) and for Yi -Wang-Liang- Ku (No. 20051020d1sj05-A).

mentioned problems. An algorithm for this method has been developed which is derived from Dijkstra’s shortest path algorithm because Dijkstra’s algorithm is not suitable for service composition, which is illustrated in other sections of this paper. Given a selection policy such as minimum cost, the algorithm can find the most appropriate services. Experiments demonstrate that this algorithm sometimes has a little worse performance but can achieve best selection.

The remainder of this paper is organized as follows. Section 2 describes a motivated example and shortcoming of Dijkstra’s algorithm for service composition. The algorithm proposed for best selection will be illustrated in Section 3. Section 4 shows experiments on the algorithm. Section 5 concludes this paper.

2 Motivations

In our previous work, Web services are modeled into ontology and represented with production rules which are constraint by some values [3], e.g. their costs. For example, if a service accepts inputs such as $I_1, I_2 \dots I_m$, and generates outputs such as $O_1, O_2 \dots O_n$, it can be expressed by the following rule:

$$I_1, I_2, \dots, I_m \xrightarrow{\text{cost } t} O_1, O_2, \dots, O_n \tag{1}$$

Additionally, DAG graph is employed to represent the service composition result. An example is illustrated in figure 1, used in the following sections in this paper.

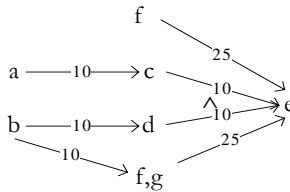


Fig. 1. An example for service composition

This example deploys some production rules in the following:

$$a \xrightarrow{10} c \quad b \xrightarrow{10} d \quad c, d \xrightarrow{10} e \quad b \xrightarrow{10} f, g \quad f \xrightarrow{25} e \tag{2}$$

In this example, edge from c to e and another from d to e are joined by symbol \wedge , which means that vertex e is generated from c and d, instead of from any one.

Let us consider problems mentioned above on the example again. When the most appropriate service must be considered and user gives a selection policy such as minimum cost, the composition plan which satisfies the selection policy has to be found and the result is illustrated by a group of paths from inputs to outputs. This is similar to problems of shortest path in Dijkstra’s algorithm. But Dijkstra’s algorithm is not suitable for composition.

Firstly, Dijkstra’s algorithm is to find the shortest path in the DAG graph. However, in the composition process, a group of paths is needed because the service

involved by expected outputs probably needs to be fed into by several services. In this paper, these paths are named group-paths and we call this multi-service problem.

Secondly, Dijkstra’s algorithm only deals with DAG whose vertex has only one element. But there may be several elements in composition, e.g. $c, d \xrightarrow{10} e$ whose vertex $\{c, d\}$ has two elements. This will cause several problems. First of all, Dijkstra’s algorithm cannot deal with it. Next, although we can easily note that $\{a, b\}$ can generate $\{e\}$, the composition system doesn’t know this because Dijkstra’s algorithm only considers two rules $a \xrightarrow{10} c$ and $b \xrightarrow{10} d$ orderly instead of together. Moreover, if $\{f\}$ generates $\{e\}$, $\{f, g\}$ will generate $\{e\}$ obviously. However, $\{f, g\}$ and $\{f\}$ are different vertexes and the system won’t know that $\{f, g\}$ can generate $\{e\}$.

To deal with these problems, we developed an algorithm for best selection derived from Dijkstra’s algorithm. Next section will illustrate the algorithm in detail.

3 Algorithm for Best Selection

Let us have a glance at the shortcoming of Dijkstra’s algorithm and consider how to handle them. If one service needs to be fed into by several services, e.g. service C needs to be fed into by services A and B where services A and B are named predecessor services, there probably exist several paths. So after the shortest path of any pair of vertexes is computed by Dijkstra’s algorithm, some actions need to be performed. The services which need to be fed into by more than one service are recorded besides of predecessor services. Then the shortest path of these services can be computed by adding up all the distances of shortest paths of predecessor services and costs between it and its predecessors. Moreover, we regard elements in a vertex as a whole such as $\{f, g\}$ is thought to be a whole in $b \xrightarrow{10} f, g$ and it is different from $\{f\}$ and $\{g\}$. But some problems will arise. Consider another production rule $f \xrightarrow{25} e$ and vertexes $\{f, g\}$ and $\{f\}$. If $\{f\}$ generates $\{e\}$, $\{f, g\}$ will generate $\{e\}$ obviously. However, $\{f, g\}$ and $\{f\}$ are different vertexes and the system won’t know the fact that $\{f, g\}$ can generate $\{e\}$. This instance is named *multi-input problem*. Additional preprocessing must be performed such as generate production rule $f, g \xrightarrow{25} e$ from $f \xrightarrow{25} e$. After these processes, Dijkstra’s algorithm can be utilized to find shortest paths in composition.

Consider the shortest path in other point of view. We define the shortest-group-path weight from s to v by equation 3. In the remainder of this paper, we also call the shortest group-path from s to v $\delta(s, v)$.

$$\delta(s, v) = \begin{cases} \min \{ \mathcal{W}(ps) : s \xrightarrow{ps} v \} & \text{if there is a group of paths}(ps) \text{ from } s \text{ to } v \\ \infty & \text{otherwise} \end{cases} \tag{3}$$

$$\delta(s, v) = \min \{ \delta(s, u) + C(u, v), \delta(s, m) + \delta(s, n) + C(\{m, n\}, v) \} \tag{4}$$

$$\delta(s, v) \leq \delta(s, u) + C(u, v) \tag{5}$$

In general, if vertex v has predecessor m , n and u where m and n produce v collaboratively while u generates v alone, the shortest group-path of vertex v from source s $\delta_{(s,v)}$ can be denoted by the equation 4.

Then we have triangle inequality illustrated in equation 5 where u represents a group of vertexes which are predecessors of vertex v and produce v collaboratively.

Theorem 1: Let $G=(V,E)$ be a weighted, directed graph with weight function $C: E \rightarrow \mathbb{R}$ and source vertex s . Then for all edges $(u,v) \in E$, we have Equation (6).

Proof. Suppose that there is a shortest group-path p from source s to vertex v . Then p has no more weight than any other group path from s to v . Specifically, path p has no more weight than the particular group-path that takes a shortest group-path from source s to vertex u and then takes edges (u,v) . \diamond

The algorithm for best selection can be developed by Theorem 1. An idea is to compute the shortest group-paths of all possible pairs of inputs and outputs. These group-paths are held for the following query. When user query arrived, the algorithm traverses all shortest group-path and selects the one which meets the need of user.

Algorithm 1 illustrates the algorithm. In algorithm 1, we firstly initialize all parameters such as $C[s][v]$ (cost between vertexes), $PU[s][v]$ (production rules used in the shortest group-path between vertexes), etc. Then the process to compute the shortest group-path from any vertex in the input of all production rules is deployed. But we can note that $\{a, b\}$ is not in all vertexes in rules in above example, so $\delta(\{a,b\},\{e\})$ is not computed in this process. Meanwhile, it cannot deal with the multi-input problem. So post processes are employed to deal with these two problems. All these processes are presented respectively in this paper.

Algorithm 1: Compute All Shortest Group-path

```

input: production list pl.
output: shortest group-paths of all pairs of vertexes.
begin
1. Let AI, AO, AI', AO' be the input, output, input
   whose number >1 and output whose number >1 of all pro-
   duction in pl respectively.
2. Initialize(pl).
3. for each entry s in AI
4.     Compute shortest path of source s.
5. update the group-path to process multi-input prob-
   lem.
6. additional process to deal with multi-service prob-
   lem.
End

```

Initialize(pl) performs initialization for the whole algorithm. It fills cost array and the array of production used for the following process. It also performs some process on multi-input problem.

Algorithm 2 generating the shortest path from s to other vertexes is similar to Dijkstra's algorithm. The difference is lines 8-9 which try to deal with multi-service problem. Lines 8-9 record the services which need to be fed into by more than one service and the predecessor services. Additionally, an array Path, which saves the production rules used in shortest group-path instead of predecessors, is employed to keep the shortest path.

Algorithm 2: Compute shortest path of source s
input: source s .
output: the array holding the shortest group-path.
begin
1. $S=\{s\}$, $D[s]=0$, $P[s]=0$, $Q=AO-S$
2. for each entry i in Q
3. $D[i]=C[s][i]$; $P[i]=PU[s][i]$
4. while $Q \neq \Phi$
5. $D[k]=\min\{D[i]:i \in Q\}$
6. if ($D[k]=\infty$) break;
7. $S=S \cup \{k\}$; $Q=AO-S$
8. for each entry i in AI'
9. if ($\{k\} \subseteq i$) $G[i].insert(s, k)$
10. for each entry i in Q
11. if ($D[i]>D[k]+C[k][i]$)
12. $D[i]=D[k]+C[k][i]$; $P[i]=PU[k][i]$
13. $Distance[s]=D$; $Path[s]=P$
end

After the shortest path of each vertex is produced in algorithm 2, we need to update the shortest group-path from it to deal with multi-input problem. Then the multi-service problem is taken into account. If more than one vertex in different production rules produces other vertexes such as $\{e\}$ collaboratively, the shortest group-path of $\{e\}$ is the sum of all the distance of its predecessor and cost between it and its predecessors. Then we have to update the shortest group-paths of vertexes which can be produced by $\{e\}$. This process is finished in line 6 of algorithm 1.

4 Experimental Result

A series of experiments were carried out to evaluate the efficiency of the algorithm for best selection in composition. We developed a prototype of the algorithm in Java. All experiments were conducted on 1GHz Intel Pentium 4 Processor with 512MB of physical memory, running RedHat Linux 9.0. In the experiment, we reviewed the efficiency by comparing two algorithms from user input aspect.

Figure 3 illustrates two composition algorithms and one algorithm for user query which is omitted in this paper due to space constraints. As shown in figure 3, execution time of composition algorithm proposed in this paper increases exponentially when more rules are involved in the composition process while the time of composition algorithm without selection policy increases exponentially only in some extent. The composition algorithm with selection policy consumes more time. Once the number of rules exceeds 400, composition algorithm without selection policy is satisfied with present rules and doesn't require any other rules because it only selects one possible group-path in some order. So time consumed is not varied finally.

Response time of the algorithm is most important for user. The response time of the algorithm proposed in this paper should be the time consumed by the algorithm for user query instead of best selection because the shortest group-path generated in algorithm for best selection can be held for following query. As shown in figure 3, response time of the algorithm for user query is less than algorithm without selection policy when the number of rules is below 420.

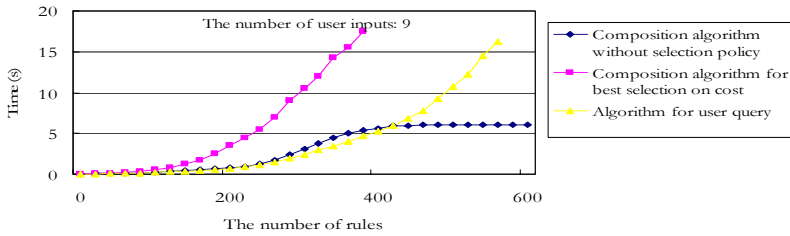


Fig. 2. Time consumed in three algorithms on the total rules

Experiments show that our algorithm spends a little less time than the algorithm without selection policy when the number of production rules is in certain extent. Once the certain extent exceeds, our algorithm consumes just a little more time which is acceptable but achieves the most appropriate composition plan.

5 Conclusion

The contributions of our work are presenting a method for computing shortest group-path in domain of service composition and developing an algorithm to implement this method. Different from the traditional shortest path, the shortest group-path in service composition involves the *multi-input problem* and the *multi-service problem*. To deal with these problems, a method derived from Dijkstra's algorithm is brought in this paper. An algorithm is developed to implement the best selection in composition process. Experiments show that the algorithm achieves the most appropriate composition plan although it sometimes spends a little more time.

References

1. Ferda Tartanoglu, etc, "Dependability in the Web Service Architecture". Proceedings of the ICSE Workshop on Architecting Dependable Systems, May 2002, Orlando, FL, USA.
2. B. Benatallah, M. Dumas, Q. Z. Sheng, H. H. Ngu, "Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web services", Proceedings of the 18th International Conference on Data Engineering (ICDE'02).
3. Jiamao Liu, Juntao Cui, Ning Gu, "Composing Web Services Dynamically and Semantically", Proceedings of 2004 International Conference on e-Commerce Technology for Dynamic e-Business(CEC-EAST 2004), September, 2004, Beijing, China.

Using Quantitative Association Rules in Collaborative Filtering

Xiaohua Sun¹, Fansheng Kong¹, and Hong Chen²

¹ Institute of Artificial Intelligence, Zhejiang University,
Hangzhou, Zhejiang, China, 310027

² Hangzhou Bell Telecommunication System Co., Ltd.,
Hangzhou, Zhejiang, China, 310007

{sunxh, kfs}@zju.edu.cn, hzbell1@hzbell.com.cn

Abstract. Recommender systems make information filtering for user by predicting user's preference to items. Collaborative filtering is the most popular technique in implementing a recommender system. Association rule mining is a powerful data mining method to search for interesting relationships between items by finding the items frequently appeared together in a transaction database. In this paper, we apply quantitative association rules to mining the relationships between items, and then utilize the relationships between items to alleviate the data sparsity problem in the neighborhood-based algorithms. The proposed method considers not only similarities between users, but also similarities between items. The experimental results on two publicly available datasets show that our algorithm outperforms the conventional Pearson method and adjusted cosine method.

1 Introduction

Collaborative filtering is an approach to make recommendations by finding correlations among users of a recommender system. The problem of collaborative filtering is to predict how well a user will like an item that he has not rated given that ratings for other items and a set of historical ratings for other users[4].

Association rule mining[2] is a powerful data mining method to search for interesting relationships between items by finding the items frequently appeared together in the transaction database. In this work, we apply quantitative association rules to mining the relationships between items, then compute the prediction on an item for an active user by computing the weighted sum of the ratings given by the user on the items similar to the target item.

The rest of the paper is organized as follows. The next section describes some related works associated with our algorithm. In Section 3, we propose our algorithm. Then, we evaluate our algorithm on two publicly available datasets in Section 4, and show that our approach outperforms the Pearson method and adjusted cosine method. Finally, we conclude the paper and discuss further development.

2 Related Works

2.1 Quantitative Association Rule

Association rules describe the relationship between the items that are frequently bought together. Agrawal et al.[2] gave a formal description of the problem of mining association rules. The problem of mining association rules is to find all association rules whose support and confidence are greater than or equal to user-defined minimum support (minsup) and minimum confidence (minconf). Quantitative association rule describes association between quantitative items or attributes[5]. In these rules, quantitative values for items or attributes are partitioned into intervals.

2.2 Pearson Approach

The GroupLens[6] system first introduced a collaborative filtering system using a neighborhood-based algorithm, it uses Pearson correlations to weight user similarity and all available correlated neighbors to compute a final prediction by performing a weighted average of deviations from the neighbor's mean.

In the Pearson algorithm, the predicted rating of the active user on item j , $P_{a,j}$, is a weighted sum of the ratings of the other users:

$$P_{a,j} = \bar{R}_a + \kappa \sum_{u=1}^n w_{a,u} (R_{u,j} - \bar{R}_u) \quad (1)$$

$$w_{a,u} = \frac{\sum_{i=1}^m (R_{a,i} - \bar{R}_a)(R_{u,i} - \bar{R}_u)}{\sqrt{\sum_{i=1}^m (R_{a,i} - \bar{R}_a)^2} \sqrt{\sum_{i=1}^m (R_{u,i} - \bar{R}_u)^2}} \quad (2)$$

Where, κ is a normalizing factor such that the absolute values of the weights sum to unity. $R_{a,i}$ represents the rating of the active user for item i and \bar{R}_a is the average rating of the active user over all items he had rated. $w_{a,u}$ denotes the Pearson correlation coefficient between the active user and neighbor u . m is the number of overlapping ratings, n is the number of neighbors.

2.3 Adjusted Cosine Approach

The item-based collaborative filtering algorithms compute the similarity between items and then to select the most similar items[7].The similarity between items i and j using adjusted cosine method is given by

$$sim(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_u)^2}} \quad (3)$$

Here \bar{R}_u is the average of the user u 's ratings.

3 Our Algorithm

Association rule mining can search for interesting relationships between items by finding the items frequently appeared together in the transaction database, thus we may utilize the relationships between items to alleviate the data sparseness problem in the neighborhood-based algorithms. We design a twofold approach based on the intuition.

3.1 Predict Using Quantitative Association Rules

During quantitative association rules mining, the numeric attributes should be discretized to satisfy some mining criteria, such as maximizing the confidence or compactness of the rules mined. We apply quantitative association rule mining on the ratings dataset, the items number per rule is set to 2, that is, we only compute the support and confidence associated with two items. We compute the confidences $conf$ of the rule $\text{Rate}(\text{item } i, \text{Inte}_{c_1}) \Rightarrow \text{Rate}(\text{item } j, \text{Inte}_{c_2})$ from mining the dataset, where Inte_{c_1} and Inte_{c_2} are two ratings intervals, we regard the confidences as the similarity associated with two items i and j . Then we compute the prediction on the confidences. The procedure of predicting using quantitative association rules is described as follows.

1. Map and discretize ratings to transactions. For each of the users we create a transaction containing all the items that he had rated in the past. We use the equiwidth binning strategy[5] here, divide the whole rating range into C equally sized intervals.
2. Mine quantitative association rules using the approach proposed in [8]. For every active user a , if he had rated item j with rating R_{aj} , let $\text{Inte}(R_{aj})$ denotes the interval that R_{aj} belongs to. We use the quantitative association rule mining algorithm to find the rules which has the form $\text{Rate}(\text{item } j, \text{Inte}(R_{aj})) \Rightarrow \text{Rate}(\text{item } k, \text{Inte}_c)$ with corresponding maximum confidence $conf_{jc}$, where Inte_c indicates interval c , $c \in \{1, 2, \dots, C\}$, then the prediction value of the active user a on the unrated item k is most likely lies in the interval

$$c_0 = \arg \max_c \sum conf_{jc} \quad (4)$$

So we set the prediction value with the midpoint of interval c_0 .

Then we get a pseudo fully filled ratings matrix to be used in neighborhood-based collaborative filtering method. Because the predicted ratings got in this step are not what we really want to present to the users, we call the resulted rating matrix as a pseudo fully filled ratings matrix.

3.2 Predict Using Neighborhood-Based Method

With the pseudo fully filled ratings matrix, we can predict the ratings of unrated items using the neighborhood-based method. We use the Pearson algorithm here. The procedure of predicting ratings using the Pearson algorithm is described as follows:

1. Compute the similarities between users: Compute the similarities between users with Eq.(2) based on the prediction ratings resulted from the quantitative association rules method.
2. Select the neighborhood: For an active user, select n user with the most similarities with him, n is the neighbors number.
3. Make predictions: Make prediction for an unrated item using Eq.(1).

The algorithm overcomes the sparsity problem by utilizing the latent relationships between items; it uses similarity measures to put more weight on users with similar preferences, has the advantages of both memory-based and model-based methods. The twofold approach utilizes the relationships between items to alleviate the data sparseness problem in the neighborhood-based algorithms. For convenience, we denote the algorithm as QAR_Pear below.

4 Experimentation

4.1 Dataset

We use two datasets, MovieLens[1] and Jester[3] to evaluate the presented approach. MovieLens is an existing movie recommendation system, the ratings are made on a 5-star scale (1 to 5). Jester is a web-based joke recommendation system, it has 4.1 million continuous ratings (-10.00 to +10.00) of 100 jokes. We split each of the two datasets to a training set and a test set, the ratio of ratings number in the training set and the test set is about 0.8:0.2.

4.2 Evaluation Metrics

We use two accuracy metrics, Mean Absolute Error (MAE) and Receiver Operating Characteristic curve (ROC) [4] to evaluate the accuracy of recommender system. The MAE E of a recommender system is evaluated by the equation:

$$E = \frac{1}{N} \sum_{i=1}^N |P_i - T_i| \quad (5)$$

where N is the total number of unrated user-item entries of a recommender system. P_i and T_i are the predicted value by the system and target value for an unrated user-item entry respectively. The lower the MAE, the more accurately the recommendation engine predicts user rating. The more the area under ROC curve, the more accuracy the recommender system is.

4.3 Result and Discussion

We have performed a number of experiments to compare the recommendation accuracy of the Pearson, adjusted cosine and QAR_Pear methods with the two datasets. For the MovieLens dataset, we set the intervals number $C=5$; for another dataset, we set $C=10$. In all the three algorithms, the neighbors number n varies from 5 to 100.

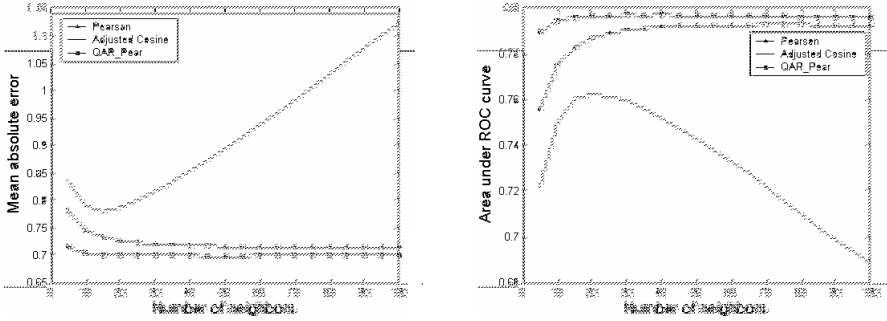


Fig. 1. The prediction results of the MovieLens dataset: MAE (left) and ROC (right)

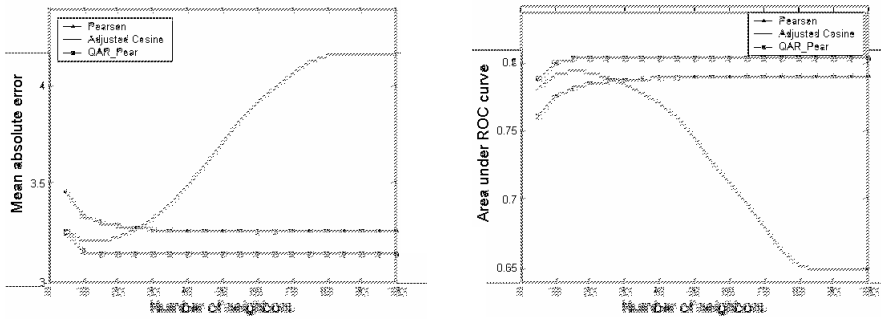


Fig. 2. The prediction results of the Jester dataset: MAE (left) and ROC (right)

Figure 1 and 2 show the results of experiments using the MovieLens and Jester datasets respectively. It can be observed from the results that the MAE of the QAR_Pear method is less than those of Pearson and adjusted cosine methods, while the area under ROC curve of the QAR_Pear method is greater than that of the other two methods. This means that the QAR_Pear method provides better accuracy than the Pearson and adjusted cosine methods on the two datasets.

From the experimental results of the two datasets, we can show that the proposed method has potential to provide better performance than the Pearson and adjusted cosine algorithms.

5 Conclusion

The proposed algorithm overcomes the sparsity problem by utilizing the latent relationships between items; it uses similarity measures to put more weight on users with similar preferences, has the advantages of both memory-based and model-based methods. Our future work will be using multi-level quantitative association rules to improve scalability and alleviate the sparsity problem further.

References

1. MovieLensdataset, <http://www.cs.umn.edu/Research/GroupLens/index.html>(2001)
2. Agrawal, R. and Srikant, R., Fast algorithms for mining association rules, In Jorge B.Bocca, Matthias Jarke, and Carlo Zaniolo, editors, Proc.20th Int.Conf.Very Large Data Bases, VLDB, Santiago, Chile (1994) 487-499
3. Goldberg, K., Roeder, T., Gupta, D., and Perkins, C., Eigentaste: A constant time collaborative filtering algorithm, Information Retrieval, Vol.4(2), (2001)133-151
4. Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J., An Algorithmic Framework for Performing Collaborative Filtering, In Proceedings of ACM SIGIR'99, ACM press (1999)
5. Jiawei Han and Micheline Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann (2000)
6. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J., GroupLens: An Open Architecture for Collaborative Filtering of Netnews, In Proceedings of CSCW '94, Chapel Hill, NC (1994)
7. Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J., Item-based collaborative filtering recommendation algorithm, Proceedings of the Tenth International World Wide Web Conference, ACM Press (2001) 285-295
8. Srikant, R. and Agrawal, R., Mining Quantitative Association Rules in Large Relational Tables, SIGMOD'96, Montreal, Canada (1996) 1-12

Extracting, Presenting and Browsing of Web Social Information

Yi Wang and Li-zhu Zhou

Tsinghua University, Beijing 100084, China
wangy01@mails.tsinghua.edu.cn
dcszljz@tsinghua.edu.cn

Abstract. We address the problem that current Web applications present mainly the content-centric information, but lack cues and browsing mechanisms for online social information. After summarizing key concepts of Web social information and analyzing characteristics of various visualization methods, we propose a set of general tools for rapid development social interfaces for interactive Web applications, including a simple but flexible programming language *VisMap* designed for Web application developers and maintainers to automate the process of extracting social information from the content-centric information, and a set of visualization methods encapsulated as *VisModules* for social information presentation and browsing. The interface of *VisModules* to *VisMap* programs are called *visual scheme*, which maps key concepts of social information to graphical elements of visual metaphors, and shields programmers from complex details of 3D graphics. Our goal of generalization is in contrast to previous works of creating specific visual metaphor for certain Web interaction.

1 Introduction

Although the Web is initially designed as a convenient way for information sharing and browsing, current technologies have evolved to turn the Web as an active arena of social activities. A number of interactive Web applications have been created for social communications, such as Web forums supporting white board, instant chat, and online game, besides traditional ways of reading and posting.

But, currently the Web is concentrating on presenting content-centric information, but lacks cues and browsing mechanisms for the online social information [1]. For example, Web forums usually list all posters, but do not show who the participants are, what the purpose, tenor, and norms of the forum are; which sorts of newcomers are welcomed; how the most active participants contribute. However, in many cases it is exactly the social information that is most valuable. The participants, especially newcomers, always want to have an overview on a Web society before they ask, post, talk, and battle. The administrators and maintainers also need statistic of the social information to help them keep online communities flourishing.

We try to overcome the problem is by extracting social information from content-centric information and then publishing both of them through the Web. This strategy is practical because for most interactive Web applications the majority of the content-centric information is stored centrally on the server side and well-structured in relational databases, which can be directly inputted to sociological analysis algorithms, such as social network analysis [2]. After extraction, we consider presentation with interactive 3D techniques for its better expressiveness, intuitiveness, and utilizing of display space than traditional Web presentation based on text and images. Furthermore, interactive graphics operations like zooming, panning and fish-eye provide a smooth and natural way of drilling the information space without losing.

Neither social analysis nor social information visualization is new. Many analysis and visualization algorithms have been proposed during the past years ([1], [2], [3], [4], [5]). But, from the aspect of information management, there is not any single analysis method or visual metaphor that is general for all kinds of social relationships embedded in all kinds of Web interactions. To address this problem, we proposed a framework (c.f. Figure 1) that support encapsulating various analysis and visualization algorithms as software modules that are callable by new light-weight programming language *VisMap*, which turns complex Web application development into organizing an information processing pipeline of social information extraction, presentation, and browsing with dozens of code lines. We believe this framework features several innovative aspects:

- *A unified tool for both analysis and presentation of social information.* *VisMap* avoids the situation in Web world where different part of a system are created with different tools by incorporating the abilities of database access, social information extraction, analysis, and presentation.
- *A platform-independent tool for Web application development.* *VisMap* is designed to be an interpretative language to avoid architectural dependencies.
- *Flexible choosing of analysis methods and visual metaphors.* Analysis and visualization algorithms are encapsulated as external software modules called by *VisMap* programs. So the developers are free to choose components that matches characteristics of their application, but be shielded from details.
- *User interaction.* *VisMap* supports an automatic GUI generation mechanism by mapping *GUI variables* defined in *VisMap* programs to GUI widgets.
- *Extensibility.* The ability of *VisMap* to invoke external modules (with *VisModule* as a typical example), it is convenient to incorporate new analysis algorithms and visualization methods into the framework.
- *Performance.* Because *VisMap* program interpreter runs on client side, the computation is naturally distributed. And with external modules implemented with compiled languages, performance of analysis and browsing can be guaranteed comparable to normal visual analysis applications. The workload of database access is in the same level as traditional content-centric Web applications.

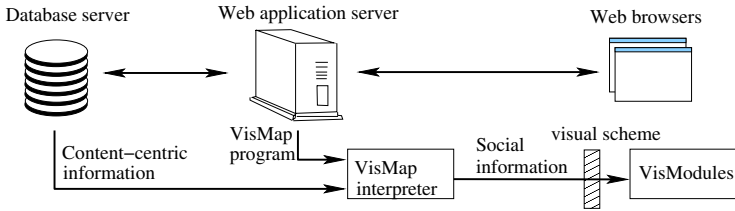


Fig. 1. Creating social interface for interactive Web applications with VisMap program and VisModule

2 The VisMap Language

The language of VisMap is designed for Web developers and maintainers to organize their choice of methods of mining and presenting social information. Except for its functionality, we try to design VisMap a simple language with learning curve as short as *HTML* and *Javascript*. In this section, we discuss the features of VisMap that makes it suitable and complete for the work.

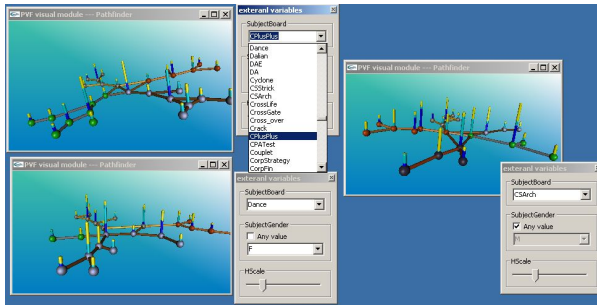


Fig. 2. Three running instances of a sample VisMap program showing co-thread relations. Each instance has a automatically generated GUI panel holding widgets allowing users to express their current interest.

- *Data type system.* As a data manipulating language, elementary types of VisMap are compatible with SQL (ISO 9075-1987). And a special first-order elementary type, *function handler* (`fhandle`), is to support *callback* from external modules. Compound types of VisMap include `tuple`, `enumeration` and `gui`. `tuple` organizes fixed number of variables in different types as `struct` in C. An `enumeration` organizes arbitrary number (can be infinity) of variables as a dynamic array. An enumeration of tuples is usually used to represent a SQL query result. `gui` is a specialization of `tuple` with certain structure to describe necessary information (e.g., the possible choices of its value represented as an `enumeration`) to map a GUI widget.

- *Embedded SQL*. To drill database to extract social information, VisMap supports embedded SQL, which can include VisMap variables directly as parameters. Syntactically, embedded SQL programs are enclosed within double braces, and treated as a remote procedure call (RPC) to database server.
- *Automatic GUI generation*. VisMap developers do not need a GUI designer, a GUI panel will be created at runtime, and all the defined gui-typed variables are mapped to automatically generate GUI widgets. An example is shown in Figure 2.
- *Function extension*. The VisMap is extensible to interact with external modules through synchronous function calling or asynchronous callback notification. A callback function defined in VisMap program can have its handle retrieved by an operator $\&$ and then passed to an external module. Given the handle, external modules can call VisMap functions to notify certain kinds of events.

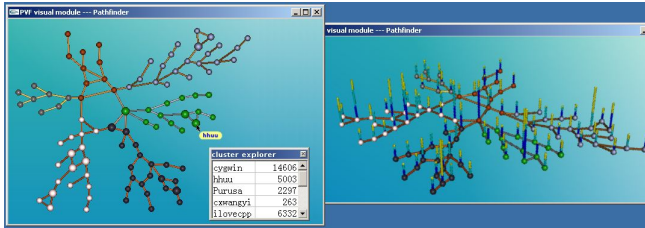
3 VisModule and Visual Schemes

Every interactive visual metaphor is designed to represent a certain kind of object and support certain behaviors. So, in order to make VisModules for VisMap, we have to abstract parameters defining the appearance and behavior of selected visual metaphors suitable for social information representation. In this section, we discuss the commonality of various visual metaphors suitable for presentation of social information, and summarize the principles of parameter abstraction as *visual scheme*.

The key elements that makes up a society include: (1) *Individuals*, as the fundamental elements of any society, carrying both *static features* recorded in database and *dynamic information* retrieved by statistics; (2) *Groups*, as a thumbnail of a number of individuals with certain commonality; and (3) *Relation*, as the glue that joint individuals or groups to form a society. We concern primarily on three typical types of social relationship. (We use screen shots in following discussion from a real VisMap project that creates social information browsing interface for a real large scale Web forum[6].)

- *Relations between individuals*. As the fundamental solid entities of societies, individuals and their relations can be intuitively visualized as a graph — with vertices corresponding to individuals and edges representing the relations. Figure 3(a) shows a VisMap program utilizing a VisModule that encapsulates the *Path-finder* network[7] to represent the *co-threading* relation between some users in the Web forum.
- *Relations between groups of individual*. For Web forums, a discussion boards represent the group of users once posted on. Most Web forums categorize their discussion board and forms a parent-child relations between the groups. Figure 3(b) use *TreeMap*[8] VisModule to show such relationship.
- *Relations between individuals and groups*. A typical relationship between individuals and groups is the contribution relation of users to discussion boards. As shown in Figure 3(c), a *Cluster-Map*[9] VisModule shows users

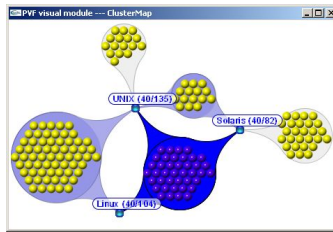
(as balls) that once posted on one of the three discussion boards (“UNIX”, “Linux”, and “Solaris” as blue balls). Users are clustered as blobs by the posting relationship with boards.



(a) Path-finder VisModule



(b) Tree-map VisModule



(c) Cluster-map VisModule

Fig. 3. Screen shots of a real VisMap project.

Based on the three summarized key concepts of social information — individual, group, and relation, we define visual schemes for each VisModule as a set of three variables `IndividualList`, `GroupList`, and `RelationList` in type of `enumeration<tuples<TypeList>>`, which are assigned by VisMap programs and then visualized as graphical elements by VisModules. For the three VisModules demonstrated above, the placeholders `TypeList` are listed in Table 1.

Table 1. The visual schemes of three VisModules

	Path-finder	Tree Map	Cluster Map
IndividuleList	int color real height	string name string photoURL	string name
GroupList		real size	string label
RelationList	int individualIndex1 int individualIndex2 real weight	int groupIndex int parentGroup	int individualIndex int groupIndex
		enumeration<int> individualIndex	

4 Discussion and Conclusion

Based on analyzing and summarizing various Web interactions and visualization methods, we proposed and implemented an ease-to-use tool for Web application developers, maintainers, and even experienced users to develop social interface as interactive visual analysis programs. As a system aimed work, there are still some aspects worthy of more considerations,

- **Interact with Web browser.** Currently, the VisModules are implemented as stand along C++ programs to ensure performance of graphics rendering and language interpretation. This makes it hard to interact with the Web browser. A possible solution is to re-implement VisModules as plugins of Web browsers, and to allocate a MIME type for the VisMap language.
- **Security.** To ensure security of the framework, besides to incorporate VisMap interpreter into the security system of Web browser, we also need an authentication mechanism to ensure the harmlessness of VisMap extensions. A possible solution is *digital signature* that has been used for Java Applets and Microsoft ActiveX controls. The security requirement on the server side can be satisfied by relative mechanisms of database systems, e.g., open a publish account for real-only operations, as the port 80 opened for real-only WWW service.

References

1. Lee, A., Girgensohn, A., Zhang, J.: Browsers to support awareness and social interaction. *IEEE Computer Graphics and Applications* **24** (2004)
2. Hanneman, R.A.: Introduction to Social Network Methods. Web publication available through <http://faculty.ucr.edu/~hanneman/> (2000)
3. Sack, W.: Conversation map: A content-based usenet newsgroup browser. *Proc. 5th Int'l Conf. Intelligent User Interfaces*, ACM Press (2000) 233–240
4. Smith, M., Fiore, A.: Visualization components for persistent conversations. *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI 2001)*, ACM Press (2001) 136–143
5. et al, S.F.: Personal map: Automatically modeling the user's online social network. *Human-Computer Interaction (INTERACT 03)*, ISO Press (2003) 567–574
6. University, T.: Smth web forum, one of the largest web forum in asia. (<http://www.smth.org>)
7. Chen, C., Paul, R.J.: Visualizing a knowledge domain's intellectual structure. *IEEE Computer* **34(3)** (2001) 65–71
8. Bederson, B.B., Shneiderman, B., Wattenberg, M.: Ordered and quantum treemaps: making effective use of 2d space to display hierarchies. *ACM Transactions on Graphics* **21(4)** (2002) 833–845
9. Fluit, C., Sabou, M., van Harmelen, F.: Cluter map — ontology-based information visualization. Chapter 3 of *Visualizing the Semantic Web*, editors Vladimir Geroimenko and Chaomei Chen (2002) 36–48

An Ontology-Based Host Resources Monitoring Approach in Grid Environment*

Yijiao Yu and Hai Jin

Cluster and Grid Computing Lab,
Huazhong University of Science and Technology, Wuhan, 430074, China
yjyu@mail.ccnu.edu.cn
hjin@hust.edu.cn

Abstract. Monitoring the real-time status of voluntary nodes is a basic task of *Quality of Services* (QoS) management in grid. The heterogeneity of distributed host resources is an obvious obstacle of grid resources monitoring. An ontology-based approach is presented in this paper to help monitor host resources, focusing on integrating and sharing the status information of grid resources. The ontology designed is originated from *Management Information Base* (MIB) in network management. The modeling methods are presented in detail, including data type translation rules, class and properties translation rules, and an ontology is proposed in this paper.

1 Introduction

As a large-scale distributed system, grid discovers unemployed resources in network, collects idle resources, and provides computing and storage services to users. If the real-time status of employed resource of all voluntary nodes can be represented in easily readable and understandable format to the load balancing subsystem of grid, the scheduler is able to assign the requests from users to the idle and appropriate nodes, and the QoS of grid is improved [1]. Due to the heterogeneity of host resources, the job scheduler of grid has difficulty in understanding and integrating the status reports from each node in various syntax formats and semantic representation.

Ontology is a powerful knowledge representations and management tool, and it is popular in the heterogeneous information management in web service, semantic web and grid applications [2]. Some grid monitoring systems have been built [3][4], however, the uniform and sharable representations of host resources are not discussed. The semantic representations, with ontology, of host resources are presented, and some encouraging results have been reported in [5]. In this paper, an ontology-based host resources monitoring approach in grid environment is proposed.

The remainder of this paper is organized as follows. Section 2 shows the modeling principles of the host resources ontology. Section 3 illustrates the translation rules from SMI to OWL. Section 4 presents the ontology. Section 5 concludes this paper with additional comments.

* This work is supported by National Basic 973 Research Program of China under grant No.2003CB317003.

2 Build the Ontology from MIBs

Ontology is an explicit specification of a conceptualization where definitions associate concepts, taxonomies, and relationships with human-readable text and formal, machine-readable axiom. It is not difficult to define some concepts and the relationships, but how to define a minimal and complete set of concepts and relationships is a challenge. The principles of our ontology modeling are listed below.

First, the concepts and properties in ontology of host resources should originate from some existing open network devices monitoring and management standards or protocols. The completeness of the concepts, properties and constraints has been discussed for a long time during the standards proposing and improving period. In addition, the concepts and properties in ontology, extracted from the existing and popular protocols, are easy to be accepted and shared by others. Due to the popularity of *Simple Network Management Protocol* (SNMP), most of the nodes in grid support SNMP agents. Up to now, more than 100 *Request for Comments* (RFC) documents, related to MIBs, have been proposed [6] and at least 15 RFC documents, with the “standard” or “proposed standard” status, are related to host resources. Therefore, it is necessary to investigate the MIBs of host resources before modeling.

Second, ontology should describe the host resources in grid completely, and the number of concepts should be as few as possible. All the applications, winners of the Semantic Web Challenge 2003, use simple ontologies, and few ontologies contain more than 100 concepts [7]. According to this experience, we check the MIBs and find that some concepts are not valuable in grid environment. Although they are useful in the traditional network management scenario, these concepts or properties are discarded during our ontology modeling.

3 Converting SMI MIBs to OWL

Although MIBs are expected to be translated into lightweight ontology only with the classes and properties in [5], the detailed methods of translating MIBs into RDF is not illustrated. The translation rules from *Structure of Management Information* (SMI) MIBs to OWL are categorized into data type, class and semantics constraints.

3.1 The Translation Rules of Data Types

Mapping data types from SMI to OWL is the first step of the host resources ontology modeling. The data structure in MIBs must be one of the following: a base type, the BITS construct, or a textual convention [8]. Furthermore, conceptual tables and conceptual rows are frequently utilized to describe complex resources. The translation rules about the four kinds of data types are illustrated.

The data types of leaf nodes in MIB tree belong to “base types”, and all kinds of data types in “base types” are listed in the first column of Table 1. There is no specific definition of data types in the latest OWL specification, and the RDF mechanism for data types is inherited by OWL [9]. OWL uses most of the built-in XML schema data types and more than 35 basic data types are recommended for use with OWL, such as `xsd:string`, `xsd:integer`, `xsd:unsignedLong`.

Table 1. The mapping rules of base data types in SMI MIBs

Data types in SMI	Domain	Data types in OWL
INTEGER	-2147483648..2147483647	xsd:integer
OCTET STRING	String (SIZE (0..65535))	xsd:string
OBJECT IDENTIFIER	0..4294967295	xsd:nonNegativeInteger
Integer32	-2147483648..2147483647	xsd:integer
IpAddress	String (SIZE (4))	xsd:string
Counter32	0..4294967295	xsd:nonNegativeInteger
Gauge32	0..4294967295	xsd:nonNegativeInteger
TimeTicks	0..4294967295	xsd:nonNegativeInteger
Unsigned32	0..4294967295	xsd:nonNegativeInteger
Opaque	OCTET STRING	xsd:string
Counter64	0..18446744073709551615	xsd:unsignedLong

If a domain of data type in SMI is identical to a domain in OWL, the two data types can be mapped directly. For example, “INTEGER” in SMI is from -2147483648 to 2147483647, while “xsd:integer” in OWL is also within the same domain. Therefore, “INTEGER” in SMI is translated into “xsd:integer” directly. Some data types in SMI can not find the exactly identical data types in OWL directly, e.g. “IpAddress”. To simplify the translation rules, these data types in OWL is selected to be the target data types of translation, which is the minimal closure of the data types in SMI. As an example, “IpAddress” is represented with string in SMI, no more than 4 characters, and mapped to “xsd:string”. This translation enlarges the representation domain of the data types in SMI. However, with specific restrictions of the domain when developing ontology, we can make them equally. Under the two principles, all the target data types of “base types” are listed in the third column in Table 1.

The BITS construct represents an enumeration of named bits. This collection is assigned non-negative and contiguous values, starting at zero. Only those named-bits enumerated may be present in a value [8]. As the translation method of “IpAddress”, the BITS construct can be mapped to xsd:nonNegativeInteger with some limitations.

3.2 Textual Convention

A textual convention is a newly-defined type defined as a sub-type of a base type. With comparison to “base types”, each of these textual convention data types has a different name, a similar syntax, but a more precise semantics. All the textual convention data types can be represented in “base types”, but the domains are limited. There are two possible ways to represent the textual conversation data types in OWL.

The first one is only to map these textual convention data types to basic data types in OWL, so necessary restrictions of domain should be attached. For example, “DisplayString” is a textual data type in SMI, which represents “OCTET STRING (SIZE (0..255))”. It is mapped into “xsd:string” with the special limitations of minimum and maximum lengths.

```

<xsd:simpleType name="DisplayString">
  <xsd:restriction base="xsd:string">
    <xsd:maxLength value="255" />
  </xsd:restriction>
</xsd:simpleType>

```

The second way is to define some new data types in OWL. Users can define new data types in OWL if necessary. For example, we define a “DisplayString” data type, whose maximum length is no more than 255, with XML schema above. In this paper, the first way is applied because the direct mapping, using the public data types, is helpful to share the application ontology with others.

3.3 Conceptual Tables, Conceptual Rows, Class and Properties

Tabular structures on an ordered collection of objects, conceptual table and row, can be regarded as complicated object types. Each conceptual table contains zero or more rows, and each row may contain one or more scalar objects, termed columnar objects [8]. An example of conceptual table, “hrStorageTable”, is illustrated below. “hrStorageEntry” is the conceptual row of this table, which depicts of a logical storage of hard disk, including the “hrStorageSize”, “hrStorageUsed”.

```

hrStorageTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF HrStorageEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        .....
    ::= { hrStorage 3 }

```

Although conceptual tables and conceptual rows are data types in SMI, they are not translated into basic data types in OWL, because the identifiers of the conceptual tables and conceptual rows are internal nodes in MIB tree. In general, the object identifier, whose data type is conceptual row, is translated into class, and multiple rows are multiple instances of the specific class. In ontology, zero or multiple instances of conceptual rows are contained in the class of conceptual table.

Classes and their properties are the basic elements of ontology. In MIB tree, the child nodes describe the parent nodes from different aspects. Usually, internal nodes should be translated into classes in ontology, but not all internal nodes should be translated into classes in the modeling. The managed objects in MIBs are grouped and organized from the view of network management and they are not completely suitable for grid resources monitoring. However, we concentrate on the dynamic device utilization status in grid monitoring, so they are reorganized in host resource ontology modeling, which will be illustrated in Section 4. In this way, the number of classes decreases dramatically during ontology modeling.

4 The Ontology

Describing the relationships between classes in ontology is difficult and necessary in ontology modeling. There are several kinds of relationships supported by OWL, such

as subclass, equivalence. To simplify the relationships of classes in host resource ontology of grid, we discard the complete path of node in MIB tree, only integrating the root nodes in the RFC documents related host resources. The relationships of classes in host resources ontology are shown in Fig. 1.

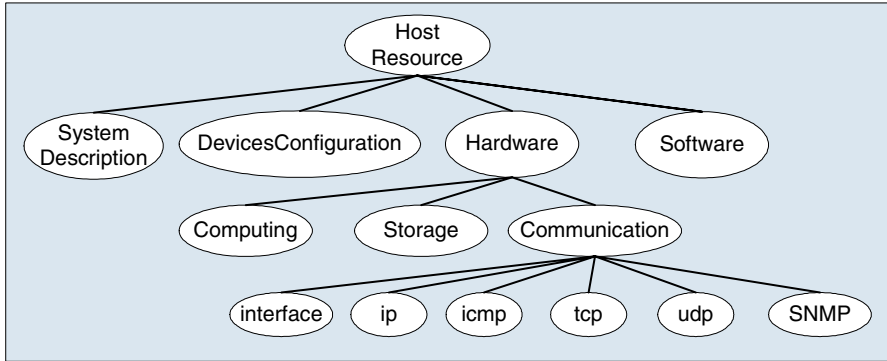


Fig. 1. Part of the host resources ontology

“HostResource” is a basic concept in the host resources ontology, which contains “SystemDescription”, “DevicesConfiguration”, “Hardware”, and “Software”. In host resource monitoring applications, the real-time status description of a voluntary node is an instance of this class. Therefore, if there are n computers in grid, there are n instances of “HostResource”. The class of “SystemDescription” gives the basic descriptions of a host, which is an aggregation of “hrSystem” subgroup in RFC 2790 and “system” subgroup in RFC 1213. All the slots of “SystemDescription” are leaf nodes in MIB tree with simple data types, e.g. “hrSystemUptime”, and “sysName”.

“DevicesConfiguration” is the static description of the hardware configuration in a host, such as the types of hard disk, printer, CPUs and video cards. The hardware configuration information is organized as child node of “hrDevice” in RFC 2790, but it is extracted as a separated class contained in “HostResources” directly. In fact, the relationships between classes in Fig. 1 are not the same as that of MIB tree. The hardware resources are categorized into computing resources, communication resources and storage resources. The values of the slots of these classes are dynamically changing when the grid nodes runs. For example, the traffic of network interfaces increases continuously and the utilization of CPUs is varying. These dynamic statuses should be monitored periodically. The MIBs of WWW, E-mail, and DBMS can be translated into classes as the MIBs of hardware resources. Moreover, the specific MIBs of grid can be defined and integrated into the ontology to provide the powerful description ability of host resources.

5 Conclusion

In this paper, we propose an ontology-based host resources monitoring approach in grid environment and describe the ontology with OWL. With the ontology, grid moni-

tors and schedulers can obtain better description, data interoperability and integration ability of the physical resources in the grid. Moreover, having host resources semantically annotated, enables us to perform semantic matching which significantly improves query results and delivers a ranked list of best matching candidates for a given service requirement from grid users. In general, the ontology-based host resources monitoring method brings a new approach to represent semantics of the host resources in grid and gives hints for the future research on semantic grid QoS.

References

1. D. A. Menasce and E. Casalicchio, "QoS in Grid Computing", *IEEE Internet Computing*, 4 (2004):85-87.
2. M. Cannataro and D. Talia, "Semantics and Knowledge Grids: Building the Next-Generation Grid", *IEEE Intelligent Systems*, 1 (2004):56-63.
3. W. Smith, "A System for Monitoring and Management of Computational Grids", *Proceedings of 2002 International Conference on Parallel Processing (ICPP'02)*, (2002):55-64.
4. B. Tierney, B. Crowley, D. Gunter, M. Holding, J. Lee, and M. Thompson, "A Monitoring Sensor Management System for Grid Environments", *Proceedings of the 9th IEEE Symposium on High Performance Distributed Computing*, (2000):97-104.
5. J. E. López de Vergara, J. E. L. Villagra, and V. A. Berrocal, "Applying the Web ontology language to management information definitions", *IEEE Communications Magazine*, 7(2004):68-74.
6. RFC INDEX, http://www.ietf.org/iesg/1rfc_index.txt.
7. M. Klein and U. Visser, "Semantic Web Challenge 2003", *IEEE Intelligent Systems*, 3 (2004):31-33.
8. K. McCloghrie, D. Perkins, and J. Schoenwaelder, RFC 2578: Structure of Management Information Version 2 (SMIv2), (1999), <http://www.ietf.org>.
9. D. L. McGuinness and F. van Harmelen, "OWL Web Ontology Language Overview", (2004), <http://www.w3.org/TR/2004/REC-owl-features-20040210>.

Domain-Specific Website Recognition Using Hybrid Vector Space Model

Baoli Dong, Guoning Qi, and Xinjian Gu

College of Mechanical and Energy Engineering,
Zhejiang University, 38 Zheda Road, Hangzhou 310027, P.R. China
tydbl@hotmail.com, gnqi@zju.edu.cn, xjgu@cmee.zju.edu.cn

Abstract. Domain-specific website recognition is a key issue for specific web resources available. The same topic websites are similar in the content structures and textual contents. According to vector space model, hybrid vector space model about website topic was proposed. This model exploited text feature instead of tree and graph ways to represent the website link structure. Its vector elements integrated text information about website content and structure characteristics extracted from relevant web pages. The topic of a website was identified through the centroid-based classification algorithm. The experiments of manufacturing-topic website recognition were implemented to verify the performances of this method. The results indicate that this model is suited to feature description of topic-specific websites. Moreover, it has good applicability of website classification on the Web.

1 Introduction

Automatic website topic recognition helps to improve the accuracy and efficiency of specific resource discovery and application. One of the most difficult problems for recognition is how to represent website topic features. Vector space model (VSM) is an effective text-based feature description method. The similarity degree of page topics is judged through vector distances. Many factors are also considered into topic analysis of web page, such as anchor text[1], page title and keyword[2], feature selection[3], ontology[4], categorized model[5], etc.

Web pages contained in a site are numerous in quantity, and the connections among pages are complicated as well. The analysis scale of website topic is completely different from web page. Ester et al.[6] summarize all pages within a site into one single feature vector based on VSM. Kriegel et al.[7] take topic frequency vector represent the number of pages about particular topic within the site. Moreover, the internal link structure of website is often viewed as a hierarchy with tree or graph structure[6, 8, 9]. These methods require complicated statistics and computations. Facing the rapid growth of network size, the suitability remains to be improved.

The website content and structure characteristics reflect its topic. To specific sites, our solution is to utilize hybrid vector space model (HVSM) representing website topic feature. In this paper, we introduce this model and briefly describe its relevant proof tests about specific site recognition.

2 Hybrid Vector Space Model

2.1 Multi-feature Model for Topic Description

The contents of same topic sites are similar in general, and the internal link structure also exist similarities obviously. Anchor texts of navigation toolbars or menus in homepages are indicative for the structure information. Different topic sites have respective structure feature terms represented by anchor texts, some examples as shown below.

- Manufacturing: profile, production, sale, after service, news, ...
- Media portal: news, entertainment, health, sports, economy, ...
- Government department: organization, policy, program, statistics, news, ...

We adopt hybrid vector space model to denote website topic feature. Its vector elements are composed of structure elements w_i and content ones w'_j , which are weighting values of structure feature term t_i and content feature term t'_j respectively. T_1 represents a structure term set about anchor texts indicating link structure information; T_2 is a content term set about keywords representing text content. The topic feature term set is the union of two sets as $T = T_1 \cup T_2$. The feature terms are mapped to vector elements as $\pi : T \rightarrow \mathbf{V}$, and website topic feature can be denoted as a vector, that is

$$\mathbf{V} = \{w_1, \dots, w_i, w'_1, \dots, w'_j\} = \{w_1, w_2, \dots, w_n\} \quad n = i + j \quad (1)$$

Here n is the total dimension of vector space, i and j is the dimension of structure and content feature respectively.

HVSM is also a kind of term-based vector expression method, but its vector elements represent different semantic information and strengthen topic feature description. Traditional VSM can be regarded as simplified HVSM, namely vector elements only include T_2 .

2.2 Weighting Selection

The weight of structure elements w_i is boolean function to reflect structure information. This approach is to let w_i present 1 if feature term t_i occurs in pages and 0 otherwise.

$$w_i = \begin{cases} 1, & \text{if } t_i > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Content element w'_j is assigned by TF-IDF weighting. The value of w_i is 0 or 1 and its dimension is relatively small. To increase the analysis ability of w_i , we take normalization TF-IDF weighting

$$w'_j = \frac{tf_j * idf_j}{\sqrt{\sum_{j=1}^m (tf_j * idf_j)^2}} \quad (3)$$

Here tf_j is t'_j occurrence frequency in pages, idf_j is defined as $\log(m/n_j)$, n_j is the number of samples which contain t'_j , m is the total amount of samples.

2.3 Feature Selection

The structure feature terms are mainly extracted from anchor texts in home-pages. Different expression habits cause the synonymous term phenomenon of same semantic links(e.g. “*about us*” and “*profile*”), and that requires to be denoted by exclusive terms. We have built a synonymous structure term library through sample statistics.

The content feature terms are achieved through statistic-based mutual information method (MI). Statistic samples are constituted of topic samples and negative ones. This method firstly chooses a quantity of preliminary terms of specialized field from the Chinese thesaurus, and then computes mutual information values of samples in different classes .

$$MI(t) = \log(P(t|C)/P(t)) \quad (4)$$

Here $P(t|C)$ is occurrence probability of feature term t in class C , $P(t)$ is occurrence probability of t in all samples. All terms are ranked according to their mutual information. To select a subset of j feature terms, the j terms with highest MI values are chosen.

2.4 Object Page Search

The principles of object page search include: (1) the page priority grade is specified as homepage>enterprise introduction>product introduction>other pages; (2) The maximum search depth of URL is limited to 2 levels of website link hierarchy ; (3) the total size of pages downloaded are defined. These can guarantee the relevant degree between object pages and website topic, and alleviate the network load and processing capacity.

As to the search path of target page, a multi hierarchy strategy of web search is adopted. Anchor texts and sitemaps help to determine the search routes inside sites. Moreover, the instance-based library of search path is set up through statistical analysis of URL expression ways of certain amounts of sample sites. The heuristic search via different routes is used to access the sites.

2.5 Classification Algorithm

Manufacturing site recognition can be typically regarded as the binary classification or filtering problem. The topics of non-manufacturing sites are comparatively dispersed and very difficult for unified description. Actually topic analysis only considers manufacturing-related feature terms. The centroid-based classification algorithm is exploited[10].

Given a set of topic samples as template set and the corresponding HVSM vector set $\{\mathbf{V}_1, \dots, \mathbf{V}_m\}$, the arithmetic mean of this vector set is taken as the center vector \mathbf{V}_c that represents topic template set.

$$\mathbf{V}_c = \frac{1}{m} \sum_{k=1}^m \mathbf{V}_k \quad (5)$$

To a new website with its feature vector \mathbf{V} , the similarity degree between \mathbf{V} and center vector \mathbf{V}_c is commonly measured using the cosine function, given by

$$Sim(\mathbf{V}_i, \mathbf{V}_j) = \frac{\mathbf{V}_i \cdot \mathbf{V}_j}{|\mathbf{V}_i| \times |\mathbf{V}_j|} = \frac{\sum_{k=1}^n w_{ik} * w_{jk}}{\sqrt{(\sum_{k=1}^n w_{ik}^2)(\sum_{k=1}^n w_{jk}^2)}} \quad (6)$$

Whether the website topic is coincident with the required topic can be determined by pre-defined similarity threshold.

3 Experiment and Evaluation

3.1 Experiment Process

To verify the recognition capability of HVSM, the recognition objects are defined as mechanical manufacturing sites. Firstly relevant pages are collected from specific sites as training samples. The feature term library is set up through page parsing and selection. Meanwhile, the feature vectors of topic templates are built. The classifier trains repeatedly to decide the suitable categorized threshold value and the size of feature dimension. Secondly, Web spiders extract new URLs from website external links, traverse inside the site and gather pages. Finally the corresponding feature vector is set up after parsing pages and matching feature terms, the site topic is judged through the classifier and predefined threshold.

3.2 Contrast Experiment

We divide the data set into 1,000 training samples and 2,100 test ones. The former are used to build the feature term library and adjust the classification threshold during the training phase. The latter contains 1,460 manufacturing-topic samples, and the other non-topic ones include government sites, media sites and etc.

The experiment is to identify manufacturing sites based on HVSM and VSM separately. The topics of test samples are determined by classification threshold T_i . The total feature dimension n is 50 and the structure dimension is 15. The indexes of classification performance are specified in [11]. Fig.1 and Fig.2 indicate the similarity distribution of test samples based on HVSM and VSM, and T_i are 0.15 and 0.1 respectively. The X-axis in the illustrations stands for cosine similar values between test samples and the center vector of topic template. The HVSM-based average similar value of topic samples is approximately 0.3~0.4, while that of non-topic samples is approximately 0.03~0.05. *recall*, *precision*, *F* for HVSM is 0.84, 0.93, 0.88 separately. The average similar values between topic sites and contrast ones are obvious in order of magnitude. *F* measure of VSM declines 10.2% compared with HVSM. The recognition performance to non-topic samples is better under HVSM.

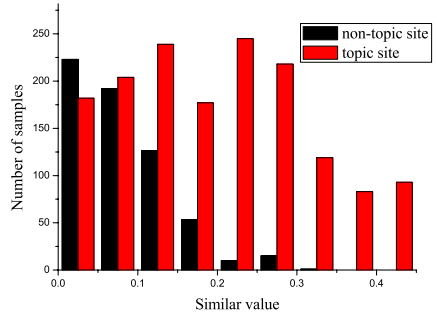
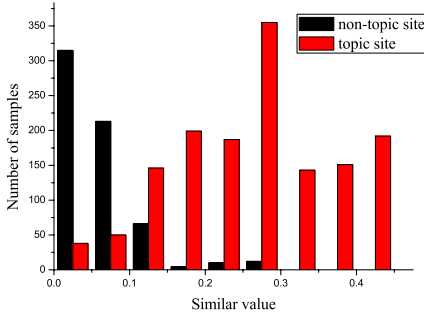


Fig. 1. HVSM-based similarity distribution **Fig. 2.** VSM-based similarity distribution

3.3 Website Recognition in Actual Internet

To assess the adaptability of HVSM in real network, 50 URLs of manufacturing-topic sites are chosen as the initial URLs for web search. New sites are found from website external links and relevant pages are downloaded. Website topic analysis is according to HVSM and VSM separately. In case new one is judged as manufacturing topic, its external links inside are extracted as new URLs for web search. Meanwhile, every site is identified by manual judgment as objective criteria. Finally, we count approximately 5,000 samples. The contrast variation curves of F measure based on two models are shown as Fig.3. Table 1 is a group of HVSM-based statistics results along the search process¹.

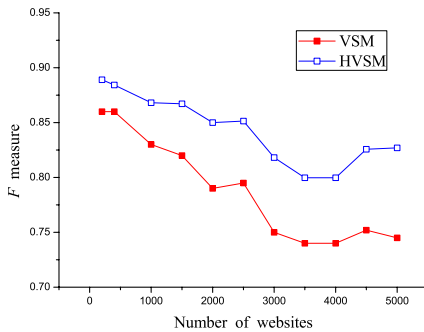


Table 1. HVSM-based classification statistics results by manual judgment

Number of site	R^+	R^-	N^+	N^-
50	50	-	-	-
100	66	5	4	25
400	213	35	19	133
1,000	294	63	28	615
3,000	906	276	95	1,723
5,000	1,598	423	213	2,766

Fig. 3. F measure variation of website classification on the Web

¹ The meanings of R^+ , R^- , N^+ , N^- are seen in [11].

Therefore, the topic recognition performance of HVSM is superior to VSM. HVSM achieves higher classification accuracy. The structure characteristics added in topic feature make for improving the recognition performance of specific sites.

4 Conclusion

This paper presents a new exploratory research on domain-specific website recognition based on hybrid vector space model. Our main contribution is HVSM-based feature description combining structure and content feature through text information. This way of feature presentation is succinct and prominent. It avoids the complexity of link structure analysis. Meanwhile, its algorithm complexity and system expense are smaller. The experimental results indicate that this model and relevant techniques contribute to improve the accuracy of topic-specific website classification, and have broad application in domain-specific resource discovery.

In our future work, we will perfect the feature term library and the search route regular library. In addition, our current research is confined to manufacturing-topic website recognition. Further researches need to be conducted about multi-class websites classification other than manufacturing sites .

References

- [1] Hersovici,M.,Jacovi,M., Maarek,Y., et al.: The Shark-search Algorithm - an Application: Tailored Web Site Mapping. *Computer Networks*. **30** (1998) 65-74
- [2] Boyan,J.,Freitag,D.,Joachims,T.: A Machine Learning Architecture for Optimizing Web Search Engines. In: Proc. the AAAI-96 Workshop on Internet-based Information Systems,Portland (1996) 334-335
- [3] Yang,Y.M.,Pederson J O.: A Comparative Study on Feature Selection in Text Categorization. In: Proc.14th International Conf. on Machine learning,Nashville (1997) 412-420
- [4] Ehrig,M.,Maedche,A.: Ontology-focused Crawling of Web Documents. In: Proc. 2003 ACM symposium on Applied computing,New York (2003) 1174-1178
- [5] Chakrabarti,S.,Dom,B.,van den Berg,M.: Focused Crawling: a New Approach to Topic-specific Web Resource Discovery. *Computer Networks*. **31** (1999) 1623-1640
- [6] Ester,M.,Kriegel,H.-P.,Schubert,M.: Website Mining: A New Way to Spot Competitors, Customers and Suppliers in the World Wide Web. In: Proc. 8th ACM SIGKDD 02,Edmonton (2002) 249-258
- [7] Kriegel,H.-P.,Schubert,M.: Classification of Websites as Sets of Feature Vectors. In: Proc. International Conference on Databases and Applications (DBA'2004),Innsbruck (2004) 127-132
- [8] Chen,X.Q.,Yu,Z.H.,Bai,S.,et al.: Automatic Information Extraction and Classification of Web Sites. In: Proc. JSCL-199,Beijing (1999) 87-92
- [9] Tian,Y.H.,Huang,T.J.,Gao,W.: A Web Site Representation and Mining Algorithm Using a Multiscale Tree Model. *Journal of Software*. **15** (2004) 1393-1404
- [10] Han,E.-H.,Karypis,G.: Centroid-based Document Classification: Analysis and Experimental Results. In: Proc. PKDD'00,London (2000) 424-431
- [11] Huang,X.Q.,Xia,Y.J.,Wu,L.D.: A Text Filtering System Based on Vector Space Model. *Journal of Software*. **14** (2003) 1538-1542

Understanding User Operations on Web Page in WISE¹

Hongyan Li, Ming Xue, Jianjun Wang, Shiwei Tang, and Dongqing Yang

National Laboratory on Machine Perception,
School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, P. R. China
{lihy, xueming, wangjj}@cis.pku.edu.cn,
{tsw, ydq}@pku.edu.cn

Abstract. Unlike Internet portals, a typical Web Information System contains lots of user operations. However, existing Web design frameworks focus exclusively on data presentation: the process of user operations is still achieved through low-level programming. That makes WIS hard to be constructed or maintained. In a Web Information System auto-construction Environment (WISE), the translation of user operations to concrete data processing code in persistent data sources is explored. This paper discusses the semantics of typical operations. The implementation code of those operations can be auto-generated with the aid of the mapping from template attributes to persistent objects.

1 Introduction

Web Information System^[1](WIS) often contains hundreds or thousands of linked Web pages with structured data and operations defined on them. WIS fetches resources by Web technologies, manipulates information to accomplish business process, and provide more openness and flexibility to various end users. Let's take a Web-based hospital information system used by Shanghai Renji Hospital in China for example. It contains 335 pages in its inpatient management subsystem. Each contains one to over ten operations. It's easy to see that data operation similarity as well as structure similarity or presentation similarity exist among pages. Constructing Web pages manually is a repeatable work, and the maintenance is ever more troublesome. WIS requires an auto-generating of those data-driven pages. But the gap between information processing on pages and underlying data sources made that a difficult job.

2 Related Work

How to construct WIS automatically becomes a challenging research field. Many methodologies contribute to that work^[2]. For instance, Relationship Management Methodology^[3], Object Oriented Hypermedia Design Methodology^[4] and UML based Web Engineering have been developed to support a hypermedia design process. They

¹ This work was supported by Natural Science Foundation of China(NSFC) under grant number 60473072.

succeed in identifying different steps to be taken in development of Web applications, but still need users to write code for data operation processing on Web pages. Most of other methodologies like UIDE^[5], Mecano^[6], Hera^[7] and template methods^[9] only concentrate on data structure or presentation similarity. None of them devote in the process and semantics of user operations even a few did process Search implicatedly.

There are two ways for auto-building of WIS: a top-down approach and a bottom-up one. The former generates Web pages according to user requirements and organizes persistent data according to page structures. [8] gives such an example. This method is easy to satisfy users, but cannot be applied in complex systems due to hard maintenance of persistent data caused by inevitable redundancy. It also fails to meet the situation when WIS must be built on a legacy source. The latter, mostly those hypermedia presentation methods, starts from persistent data design usually with the help of a DBMS, and then generates Web presentation of data. But it can hardly support auto-process of arbitrary user operations on Web pages since business data is organized by a database designer and often not for the convenient use of end users.

WISE tries a trade-off method: Web pages are generated by template model which includes definitions of structured data and operations. If a data source is present, no matter it is a legacy system or not, a mapping strategy is adopted to bridge the gap between Web applications and persistent data. After all template items map to a data source, the general user operations can map to concrete operations on that data source. The operation mapping method forms a part of our WIS auto-construction framework and is used to generate operation processing code.

3 Template Operations

This paper will discuss the built-in operations in WISE: Link, Search, Clear, Modify and Add. Link is defined on a sub template attribute or a component of a template instance. It's used for navigation among templates.

Let T be a template and $\text{Attrs}(T)$ be the set of attributes of T , Search, Clear and Modify could be defined on two disjoint subsets of $\text{Attrs}(T)$. One is used for receive user input and generates operation constraints and denoted by $CA_{S/C/M}$. The other is used for receive output from underlying persistent data source and denoted by $TA_{S/C/M}$. TA_C also generates Clear candidates and TA_M receives new value input by users. Add could be defined on only one subsets of $\text{Attrs}(T)$, which is used for receive input from users and generates new instances and denoted by TA_A .

4 Mapping Template Attributes to DBMS

Because of the structure similarity between template and nested table, one could regard template as a logic view. That viewpoint makes mapping template attributes and operations to data source possible. On the other hand, data source often use DBMS to manage business data. Those data is organized according to database storage strategy, not to template scheme. So mapping template attributes and operations to data source is necessary.

Given a template T and an attributes A of T , if A has a source D in data source M , we say D is the origin of A in M . Otherwise, A has no origin in M . If A has an origin D in M , and D can be presented as such a SQL statement: $\text{SELECT } S \text{ FROM } T_1, \dots, T_n$ ($n \geq 1$)[$\text{WHERE } W$], we say A has a basic origin in M . Let $OS_M(A)$ be the target expression in Select clause, $OF_M(A)$ be the set of tables(or views) in From clause, and $OW_M(A)$ be possible conditions in Where clause. $OS_M(A)$, $OF_M(A)$ and $OW_M(A)$ is called the original target, original range and original condition of A in M respectively.

An attribute may have an origin but not a basic one in a data source. To find basic origins of all template attributes is a preliminary work for the operation process.

Constraint 1. Given a template T , $\forall o \in \{\text{Search, Add, Modify, Clear}\}$ and $\forall A \in CA_o \cup TA_o$, if o could be converted to an operation on data source M , A has a basic origin in M .

The process of Link is much simpler than the rest four built-in operations. The process of Search is similar to the query conversion method describe in [11]. So the rest of this paper only presents the disposal of Add, Clear, and Modify.

5 Add

Let $TA_A = \{A_1, \dots, A_n\}$ ($n \geq 1$), for $\forall A_i (1 \leq i \leq n)$, the value of A_i would be inserted into one table in a data source. So $|OF_M(A_i)| = 1$ is met. Since not all data manipulations defined on views could be executed successfully in DBMS, the following constraint is made.

Constraint 2. Given a template T , $\forall o \in \{\text{Add, Modify, Clear}\}$, and $\forall A \in CA_o \cup TA_o$, if o could be converted to an operation on data source M , $OF_M(A)$ is a certain table.

In traditional DBMS, Insert, Update and Delete could process only a single table each time. But in WIS, users need to define operations mapping to a process on multiple tables at a same time. So the original ranges of different Add target may be different. The following steps explain in detail how to process Add operation.

Step1. Arrange all attributes in TA_A into m groups: the original range of each group is the same, while the original ranges of different groups are different.

Step2. Let $T_i (1 \leq i \leq m)$ be the original range of i -th group. Collect all Key-Foreign Key Constraints(K-FKC) among tables T_1, \dots, T_m from data source M . Generate the constraint dependency graph $G_M(\text{Add})$:

- (1) For $\forall T_i (1 \leq i \leq m)$, there's a certain vertex V_{T_i} in $G_M(\text{Add})$ mapping to it.
- (2) If T_j 's foreign key references T_i 's key, add directed edge from T_i to T_j .
- (3) If $\exists T_1, \dots, \exists T_k (k \geq 1)$ and $T_i \neq T_1 \neq \dots \neq T_k \neq T_j \wedge T_i \rightarrow T_1 \wedge \dots \wedge T_k \rightarrow T_j$, from $p=1$ to k , add vertex V_{T_p} into $G_M(\text{Add})$ if V_{T_p} is not in $G_M(\text{Add})$. Add directed edge from V_{T_i} to V_{T_1}, \dots , and from V_{T_k} to V_{T_j} if that edge is not in $G_M(\text{Add})$.

Step3. If there's no loop in $G_M(\text{Add})$, insert into those tables mapping to zero in-degree vertexes firstly. Suppose T_i is such a table. It's also the original range of i -th group $\{A_{i1}, \dots, A_{ik}\}$. Those attributes form the scheme of Add instance $[A_{i1}, \dots, A_{ik}]$ which could be converted to T_i 's insert scheme $[OS_M(A_{i1}), \dots, OS_M(A_{ik})]$. User input values of A_{i1}, \dots , and A_{ik} form T_i 's insert instance $(V_{Ai1}, \dots, V_{Aik})$. Now we get the SQL statement: $\text{INSERT INTO } T_i (OS_M(A_{i1}), \dots, OS_M(A_{ik})) \text{ VALUES } (V_{Ai1}, \dots, V_{Aik})$.

Repeat this step till all tables mapping to zero in-degree vertexes could be inserted.

Step4. Check a $p(p \geq 1)$ in-degree vertex V_T in $G_M(Add)$. T could be inserted if all tables mapping to V_T 's preceding vertexes have been inserted correctly. Let $V_{T_i}(1 \leq i < p)$ be such a preceding vertex. According to Step2, we know T has a foreign key FK_i referencing T_i 's key K_i . If T is the original range of j -th group $\{A_{j1}, \dots, A_{jk}\} (k \leq n)$, T 's insert scheme is $[OS_M(A_{j1}), \dots, OS_M(A_{jk}), FK_1, \dots, FK_p]$ and insert instance is $(V_{A_{j1}}, \dots, V_{A_{jk}}, V_{FK_1}, \dots, V_{FK_p})$. Here V_{FK_i} is the component value agrees in K_i for T_i 's insert instance. If T 's not a original range of any group, V_T must be a vertex added in Step2 (3). T 's insert scheme is $[FK_1, \dots, FK_p]$ and insert instance is $(V_{FK_1}, \dots, V_{FK_p})$.

Repeat this step till all tables mapping to all vertexes in $G_M(Add)$ could be inserted.

If there's a loop in $G_M(Add)$, all tables mapping to vertexes in that loop couldn't be inserted until at least one K-FKC on those tables is disabled. Then one could follow Step3 and Step4 to fulfill Add operation.

6 Clear

Step1. Clear condition is generated according to input value received by CA_C , while TA_C will be used for receive outputs from data source. This sub process is similar to Search process except that Constraint2 should be abided. We call it the preceding search of Clear operation and produce a select statement S for it.

When a Clear map to a table's deleting, the process is very simple. Is it possible to define Clear involves multiple tables? Let's take Fig.1 for example. Suppose the origins of t_1 and t_2 of template T in data source M are: $OS_M(t_1)=A.a_1$, $OF_M(t_1)=A$, $OS_M(t_2)=B.b_2$, and $OF_M(t_2)=B$. The tuples of table A and B are shown in Fig.1(a) and (b). If there's a join path $A \rightarrow B$ and the corresponding join condition is $A.a_2=B.b_1$, executing the preceding search, we'll get Clear candidates as shown in (c). It's easy to see that not every candidate could map to a unique tuple of A or B , and vice versa.

(a) Table A		(b) Table B			(c) Template T		(d) Template T	
a_1	a_2	b_1	b_2	b_3	$t_1(A.a_1)$	$t_2(B.b_2)$	$t_1(A.a_1)$	$t_2(B.b_2)$
1	2	2	6	1	1	6	1	6
3	4	2	5	2	1	5	1	5
		4	8	1	3	8	3	8
		4	8	2			3	8

Fig. 1. An example for Clear processing

When one candidate is mapping to multiple tuples of a single table, template instance clear brings on the uncertainty in tuple delete. For example, (3,8) of T shown in (c) may be generated from the join of (3,4) in A and (4,8,1) in B or (4,8,2) in B . If (3,8) is cleared, shall we delete only (4,8,1) or (4,8,2), or delete them both? That problem is caused by the filtering of duplicate tuples in query result in DBMS. The next Step will make some addition into the process of preceding search to avoid it.

Step2. If there's only one table in the range of S , jump to Step3. Otherwise, let the range be $\{T_1, \dots, T_k\} (k \geq 2)$, and the set of targets be TS . For $\forall T_i (1 \leq i \leq k)$, check if TS

contains T_i 's key. If not, add one key into TS compulsively. Or add the rest attributes of T_i which haven't belonged to TS in case there's no key defined on T_i .

Step3. Execute statement S or the evolved statement generated by Step2. Project query result on all attributes in TA_C without filtering of duplicate tuples. Now we get clear candidates on T. As shown in Fig.5(d), one could regard a (3,8) is generated from the join of (3,4) in A and (4,8,1) in B, another from the join of (3,4) and (4,8,2).

Let the set of Clear instances be $V_C = \{v_1, \dots, v_n\} (n \geq 1)$. For $\forall v_j (1 \leq j \leq n)$ and $\forall T_i (1 \leq i \leq k)$, v_j could map to a unique tuple of T_i (denoted by v_{ji}) with the help of T_i 's key. But when multiple candidates map to a single tuple, instance clear also brings uncertainty on tuple delete. Consider (1,6) and (1,5) of T, they all map to (1,2) in A. If users only clear (1,6) or (1,5), shall we delete (1,2) from A? If we do so, another instance will never be seen in T. If we don't, the clear instance exists still.

Constraint 3. Given a template T, its Clear operation could be converted to delete operation on data source M if there's only one table in original range of its preceding query, or, let the set of clear instances be V_C and the set of clear candidates be VC_C , for $\forall v_1 \in V_C$ and $\forall v_2 \in (VC_C - V_C)$, v_1 and v_2 are generated from different tuples.

Step4. Check if Constraint3 is met. If so, the clear instances in V_C could be converted to tuple deletes from tables in $\{T_1, \dots, T_k\}$. As for $\forall T_i (1 \leq i \leq k)$, the SQL statement is DELETE FROM T_i WHERE K_i IN= $\{v_{1i}(K_i), \dots, v_{ni}(K_i)\}$.

Some tables may be added into original range because of the involvement in certain join paths. For example, the join path between table A and B in Fig.1 would be $A \rightarrow C \rightarrow B$ instead of $A \rightarrow B$ and the corresponding join condition be $A.a_2 = C.c_1$ and $C.c_2 = B.b_1$. Therefore, C is added into the original range of preceding query. But no attribute of C is in TA_C . Following above steps, on delete on C is generated. But if there's K-FKC between A and C and ON DELETE CASCADE option is used in the constraint definition, referenced tuple delete from A would trigger referencing tuple delete from B. That process will be auto-executed by DBMS.

7 Modify

Modify is also taken on the result of a preceding search. In order to support Modify mapping to updates on multiple tables, tuple update uncertainty is eliminated by the same method described in Clear when one candidate mapping to multiple tuples. But when multiple candidates map to a single tuple, Constraint3 is inadequate to eliminate the uncertainty. Let's consider (1,6) and (1,5) in Fig.1(d). They all map to (1,2) in A. Constraint3 ensures (1,6) and (1,5) will either be updated simultaneously or not be updated at all. But even if they are updated simultaneously, one may change (1,6) to (3,6) and (1,5) to (5,5). The update semantics on (1,2) wouldn't be consistent.

Constraint 4. Given a template T, Modify could be converted to update operation on data source M if there's only one table in original range of its preceding query, or, let the set of Modify instances and candidates be V_M and VC_M , for $\forall v_1 \in V_M$ and $\forall v_2 \in VC_M$, v_1 and v_2 are generated from different tuples.

In the conversion of Modify to update on data source M, the first three steps are similar to that of Clear. The last step is presented as the following:

Step4. Let the set of Modify instances be $V_M = \{v_1, \dots, v_n\} (n \geq 1)$. Let all attributes for v_j in T consist subset $\{A_{j1}, \dots, A_{jm}\} (m \geq 1)$, and the original range of $\{A_{j1}, \dots, A_{jm}\}$ be $T_i (1 \leq i \leq k)$. v_j could map to a unique tuple of T_i (denoted by v_{ji}) with the help of T_i 's key. Let user input value on $A_{jo} (1 \leq o \leq m)$ in v_j be $New(v_j(A_{jo}))$. Now check if Constraint4 is met. If so, the instance Modify in V_M could be converted to tuple updates on all tables in $\{T_1, \dots, T_k\}$. As for $\forall T_i (1 \leq i \leq k)$, the SQL statement is:

```
UPDATE Ti SET OSM(A11)=New(v1(A11)),...,OSM(A1m)=New(v1(A1m)),...,
        OSM(An1)=New(vn(An1)),...,OSM(Anm)=New(vn(Anm))
WHERE Ki IN={v1i(Ki),...,vni(Ki)};
```

8 Conclusions and Future Work

In WISE, those built-in operations are converted successfully to data source managed by DBMS. WISE also provides a set of visual tools^[10] to define data structure, mapping, user operation, etc. WISE can transform those definitions to executable code under the restriction of operation semantics and fulfill database access.

There are many works to be done in the future, such as the expression of complex conditions for operations on Web pages, the support of user defined operations, dynamic operation modification, etc.

References

1. T. Isakowitz, M. Bieber, F. Vitali. Web Information Systems. Communications of ACM, ACM Press, Vol 41(7) (1998), 78-80
2. Peter Barna, et al. Methodologies for Web Information System Design. International Conference on Information Technology: Computers and Communications, (2003) 420-425
3. T. Isakowitz, et al. RMM: A methodology for the design of structured hypermedia applications. Communications of ACM, ACM Press, Vol.38(8) (1995) 34-44
4. Daniel Schewabe, et al. OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW. ACM SIGWEB Newsletter, Vol.8, Issue2, (1999) 18-34
5. J. Foley, et al. UIDE: An Intelligent User Interface Design Environment. In J. Sullivan and S. Tyler (eds), Intelligent User Interfaces, (1991) 339-384
6. A. Puerta. The Mecano Project: Comprehensive and Integrated Support for Model-Based Interface Development. Computer-Aided Design of User Interfaces, (1996) 19-36
7. Geert-Jan Houben. HERA: Automatically Generating Hypermedia Front-Ends for Ad Hoc Data from Heterogeneous and Legacy Information Systems. In Third International Workshop on Engineering Federated Information Systems, Aka and IOS Press (2000)
8. Ming Xue, Hongyan Li. Managing User Interaction Forms on Web Pages: A Template-Based Approach. Journal(Natural Science) of Peking University, Vol.40(3) (2004) 473-479
9. Jim Challenger, et al. A Publishing System for Efficiently Creating Dynamic Web Data. Proceedings of IEEE INFOCOM 2000
10. Lv-an Tang, Hongyan Li, et al. PODWIS: A Personalized Tool for Ontology Development in Domain Specific Web Information System. APWEB2005, 680-694
11. Hongyan Li, et al. An XML Based Electronic Medical Record Integration System. In : X. Sean Wang, et al. (eds): WAIM 2001, 160-167

Two-Phase Exclusion Based Broadcast Adaptation in Wireless Networks*

Keke Cai, Huaizhong Lin, and Chun Chen

College of Computer Science, Zhejiang University,
Hangzhou, China, 310027
{caikeke, linhz, chenc}@zju.edu.cn

Abstract. The basic idea of popularity-based wireless broadcast is to broadcast data with most requests. However, client requests cannot always reflect their entire data requirements. It thereby leads to the inaccuracy of broadcast and the increased number of client requests. In this paper, we propose a correlation-based broadcast scheme, in which relationships among data are used as a reference in dealing with exclusion of unpopular data. In this broadcast scheme, data to be excluded are initially divided into two parts. The popularity of data in the second part is inferred by the actual popularity of data in the first part. This is possible if a causal relationship exists in their access patterns. The results from extensive simulation experiments conclude that the correlation-based broadcast scheme can significantly improve the mean response time and reduce the number of missing requests.

1 Introduction

The overwhelming information makes it a critical issue how to realize efficient data delivery in the wireless environment. Experiments show that broadcast can better adapt to the special constraints of mobile computing environments, and therefore provide better performance. In this paper, we mainly deal with the problem of how to update the broadcast content.

The broadcast schedule, which determines what should be broadcasted and when [1], directly determines the performance of broadcast. In the past few years, many broadcast schemes have been proposed [2], [3], [4], [5]. In [6], Stathatos introduced a dynamic hybrid broadcast model. This model automatically decreases the popularity of broadcasted data, and then removes data with popularity lower than the predefined threshold. For the estimation of data popularity, this model inevitably brings the “broadcast misses”, and subsequently induces the large increase of client requests. To remedy this deficiency, a solution called “temperature probing” was proposed by [6]. The main idea is to set a limited observation time, during which data will be called back quickly once it is found popular. The method limits the number of client requests

* Supported by the Natural Science Foundation of Zhejiang Province, China (Grant no. M603230) and the Research Fund for Doctoral Program of Higher Education from Ministry of Education, China (Grant no. 20020335020).

to some degree, but frequent removal and withdrawal can adversely degrade system performance. How to identify the popularity of data being broadcasted and reduce the improper exclusions effectively is then the focus of this paper.

In this paper, we propose a two-phase exclusion model, which initially divides data scheduled for exclusion into two parts. Data in the first part will be excluded in the first instance, and shortly will be either withdrawn or discarded completely according to the state of client requests later on. Their disposals will further influence the go or stay of data in the second part. Generally speaking, factors used for data unpopularity estimation involve not only data popularity, but also data correlations implicit in client requests. It is believed that some data are likely to be popular if some other data are popular. The primary rationale is that some data are always requested together in most cases.

The rest of this paper is organized as follows. Section 2 introduces the main concepts and the proposed algorithm. The experimental study and results are presented in Section 3. Section 4 concludes the paper.

2 Correlation-Based Broadcast Algorithm

2.1 Related Definition

The most important knowledge throughout our scheme is the correlations between different data. This information, expressed by the form of association rule, can be identified by the association rules mining technique [7]. An association rule is an implication of the form $FD \rightarrow SD$, FD and SD are two data sets satisfying $FD \cap SD = \phi$. Its confidence shows the occurrence probability of FD and SD , and is defined as:

$$\text{conf}(FD \rightarrow SD) = \frac{\text{sup}(FD \cup SD)}{\text{sup}(FD)}. \quad (1)$$

For a transaction set T and a data set S , function $\text{sup}(S)$ describes the support of S by T . Let U be all the transactions of T that contain S , $\text{sup}(S)$ is defined as:

$$\text{sup}(S) = |U| / |T|. \quad (2)$$

2.2 Update of Broadcast Data

Data in the broadcast queue should be updated dynamically, which correspondingly involves two basic operations, insertion and deletion. The insertion can be easily achieved according to the observation of popular data in client requests. However, for the problem of data deletion the key is the perception of usability of broadcast data. In our paper, we adopt the prediction-based solution such as [6]. Each data for broadcast is assigned with an importance mark indicating its popularity to clients. It is initialized to the data support, and then gradually decreases by a certain proportion along with each broadcast cycle. Data in the broadcast are deemed invalid if their importance marks are lower than a certain baseline.

2.3 Two-Phase Exclusion Scheme

In [8], we utilize the correlations between inserted and deleted data and propose the correlations based broadcast scheme. In our paper, we further study the inner correlations of deleted data and propose a two-phase exclusion scheme.

The primary operation in our scheme is the partition of data that are initially scheduled for exclusion. Based on the general association rules extracted from client request history, we partition the data to be excluded initially into two parts. They satisfy certain relevancy requirements. The following algorithm *Data_Division* describes the division process. In the algorithm we always gives priority to rules that can educe larger consequent data with few antecedent data since these rules always realize the corresponding relevancy identification earlier.

Algorithm

```

Data_Division(ed, firstpart, secondpart, ar){
  sort ar in an ascending order by the number of the
  consequent data of each rule. If two rules have same
  length, rule that has shorter antecedent has the
  priority;

  for each r in ar{
    if(non-overlap(anteced(r), secondpart)){
      add data of anteced(r) to firstpart;
      for each data d of conseq(r)
        if(!belong(d, firstpart))
          add d to secondpart;
    }
    else if(belong(anteced(r, secondpart) &&
      |anteced(r)| < |conseq(r)|
      && !belong(conseq(r), firstpart))){
      remove data of anteced(r) from secondpart;
      add anteced(r) to firstpart;
      add conseq(r) to secondpart;
    }
  }
  for each data d of ed
    if(!belong(d, firstpart) &&
      !belong(d, secondpart))
      add d to firstpart;
}

```

Based on the data divided above, the two-phase exclusion is implemented to realize the final disposal of exclusion. In the first phase, data belonging to the *firstpart* are firstly excluded. For simplicity, we define the probing interval *pt* as half of the broadcast cycle. During *pt*, client requests for data of *firstpart* are observed, and *firstpart* are finally subdivided into two parts of *upd* and *wpd*, respectively containing data actually unpopular and popular. Data in *upd* are discarded completely, but data in *wpd* are re-called back to broadcast successively. In the second phase, based on the association rules existing between *wpd* and *secondpart*, data in *secondpart* and related to data in *wpd* are identified. Given the data set X , $X \subseteq wpd$, if there exists an association

rule formed as $X \rightarrow Y$, then data simultaneously contained by Y and *secondpart* are considered as the related data of X . According to our scheme, these data are kept in broadcasting, while other data in *secondpart* still need to be excluded.

3 Experimental Study

In our experiments, we have built a simulation model of the proposed system. All transactions representing client requests are simulated using IBM synthetic association data generator [9]. For each database, transaction number varies from 10k to 1M per dataset, and each transaction contains 10 items on average. In order to simulate dynamic workloads, an s -sized sliding window w is applied to the data collection. Data contained in w are simulated as the requests issued by clients in a certain period of time.

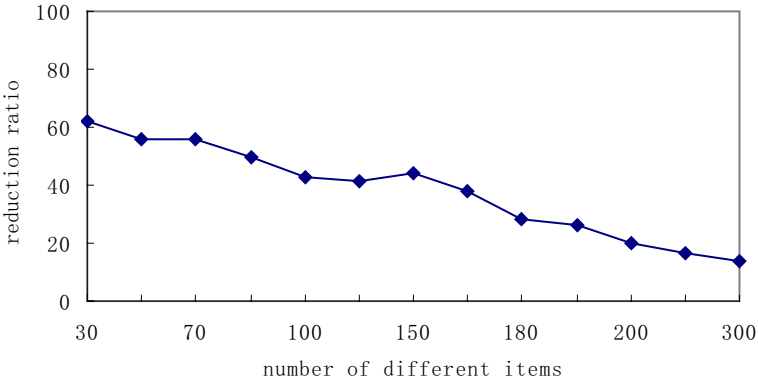


Fig. 1. Reduction of client requests with the different number of client items

In our experiments, we defined three basic performance metrics. Let ed be data to be excluded in the i^{th} broadcast. They are initially divided into two parts *firstpart* and *secondpart*. We suppose that data in *firstpart* but proved popular are denoted as wd , data in the *secondpart* and actually popular are denoted as spd , data in *secondpart* that could be inferred by wd are described as rd , and data in rd that are actually popular are denoted as rpd , then $Racc$ can be defined as:

$$Racc = |rpd| / |rd|. \quad (3)$$

$Rcmp$ is defined as:

$$Rcmp = |rpd| / |spd|. \quad (4)$$

$Rrdc$ is defined as:

$$Rrdc = |rpd| / |ed|. \quad (5)$$

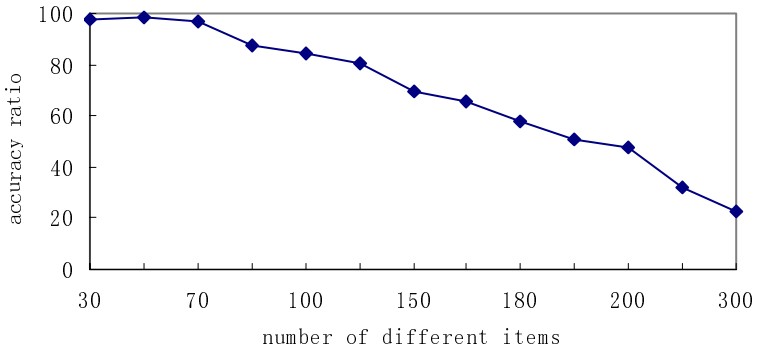


Fig. 2. The accuracy ratio of the association inference

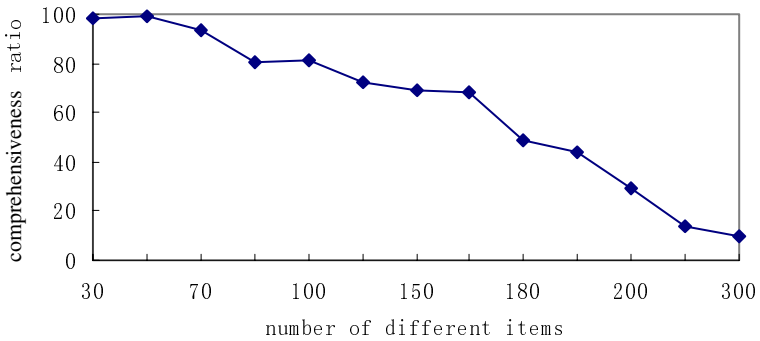


Fig. 3. The comprehensiveness ratio of the association inference

Figure 1 illustrates the experimental results of *Rrdc* with varied number of different items. From figure 1, we found out that our scheme performs best when the number of items varies from 30 to 70. It is able to reduce about 60 percent of client requests. However, with the increased number of involved items, the performance takes on a very clear downtrend. When the number of items increases to 300, the *Rrdc* only reduces about 20 percent requests. It happens mainly due to no available associations. Obviously, the more number of items a data set has, the more randomness of the generated data transactions is. If data in client requests are weakly correlated, little regularity can be referenced.

Fig. 2 and fig. 3 respectively illustrate the experimental results of *Racc* and *Rcmp*. The two figures similarly prove that the experiments perform well if there exists sufficient association information. We thereby concluded that if data contained in client request transactions are not stochastic, but imply some special relevancies, their relevancies information represented in the form of association rule could significantly improve the accuracy of popularity identification.

4 Conclusions

In this paper, we propose an efficient method for popularity identification while adapting the contents of the broadcast. The basic idea is that, for some special data that are being broadcasted, their popularity can be inferred based on the valued associations existing in client requests. The most popular data will continue being broadcasted. The experiments prove that with the existence of valued correlations, our method avoids a large number of unnecessary data omissions and thereby reduces the overall client requests.

References

1. J. Xu, D. L. Lee, Q. Hu, and W.-C. Lee, Data Broadcast. Handbook of Wireless Networks and Mobile Computing. Chapter 11, Ivan Stojmenovic, Ed., New York: John Wiley & Sons, ISBN 0-471-41902-8, Jan (2002) 243-265
2. Acharya, S., Franklin, M., Zdonik, S, Disseminating Updates on Broadcast Disks. Proceedings of 22nd VLDB Conference. India (1996)
3. Datta, A., Celik, A., Kim, J. and VanderMeer, D.E., Adaptive Broadcast Protocol to Support Power Conserving Retrieval by Mobile Users. Proceedings of 13th International Conference on Data Engineering. (1997)
4. Swarup Acharya, Rafael Alonso, Michael Franklin, and Michael Zdonik, Dissemination-based Data Delivery Using Broadcast Disks. IEEE Personal Communications Magazine, 2(6), December (1995)
5. Demet Aksoy, Michael J. Franklin, Stanley B. Zdonik, Data Staging for On-Demand Broadcast. VLDB. (2001) 571-580
6. K. Stathatos, N. Roujssopoulos and J.S. Baras, Adaptive data broadcast in hybrid networks. VLDB Conference. (1997) 326-335
7. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In Buneman, P., Jajodia, S., eds.: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data. Washington, D.C. (1993) 207-216
8. K. Cai, H. Lin, C. Chen, Correlation-based Data Broadcasting in Wireless Networks. BNOCD. (2005)
9. IBM Quest Data Mining Project, Quest Synthetic Data Generation Code. <http://www.almaden.ibm.com/software/quest/Resources/datasets/syndata.html#assocSynData>. (2005)

Web Service Collaboration Analysis via Automata*

Yuliang Shi, Liang Zhang[†], Fangfang Liu, Lili Lin, and Baile Shi

Department of Computing and Information Technology, Fudan University, China
{031021056, zhangl, 041021055, 042021120, bshi}@fudan.edu.cn

Abstract. It is evidenced that formal analyses are helpful for web services interactions. However, most current web services choreography proposals, such as BPEL4WS or WSCI, only provide notations for describing the message flows in web service collaboration, lacking of reasoning mechanisms to verify the process of interacting among them. In this paper, we present a formalization of web services interaction based on WSCI using the approach of automata. The method can check whether two or more web services are compatible or not in their collaboration.

Keywords: Web services, WSCI, Automata.

1 Introduction

Recently, web services have emerged as a new paradigm that supports loosely coupled distributed software systems. However, in most cases, a single web service can hardly satisfy practical applications. For a complex business process, collaboration of multiple web services is required to achieve the preset goal. At present, several proposals, such as BPEL4WS^[1] and WSCI^[2], have been used to describe the process of collaboration. But most choreography proposals only provide notations to describe the message flow in the collaboration, lacking of some reasoning mechanisms to verify the interaction between web services. As a result, a predefined process is likely to behave abnormally due to web services interacting incorrectly. To deal with this issue, it is needed to provide a formal method to verify the soundness of underlying web services' collaboration.

Frankly, there exist several verification methods^[3-9]. They are based on the global information interaction between web services. The main task is to find out whether or not the interaction process matches the given model, which can verify the correctness of global conversation between services. However, these verification methods don't take it into account whether the behaviors of a service are compatible with the global information interaction. Consider two web services, one for online shopping and the other for customers. Suppose that they are syntactically compatible in invoking interfaces, but not negotiate well in collaboration protocol. The online shopping waits

* This work is partially supported by the National Basic Research Program (973) under grant No. 2005CB321905, the Chinese Hi-tech (863) Projects under the grant No. 2002AA4Z3430, and No. 2002AA231041

[†] Correspondence author.

for payment before sending the product while the customer believes Cash-on-Delivery. It is certain that the collaboration between them will lead to a deadlock, no matter how well they are compatible in invoking interfaces. Thus, for given global information interaction which has been verified correctly by above methods, there still might exist a web service of which the behaviors are not compatible with it. This collaboration will fail finally in practice.

In this paper, we propose a verification method of collaboration between WSCI-based web services. Here we employ a formal method, say automata, to formalize WSCI. Based on the automata, we can verify the compatibility between the behaviors of web services through a client/server model. Such compatibility is important to guarantee the successful completion of collaboration of web services.

The paper is organized as follows. After this introduction, section 2 surveys related works. In section 3, we discuss how to model web services based on WSCI with DFA. Section 4 describes how to analyze the process of collaboration, and followed summary and future works.

2 Related Work

Recently, verification of web service collaboration has been attracting much attention. The main methods include MSCs, FSA, logic, Petri net etc. In [3], a model-based approach is proposed by the authors for verifying Web service composition, using Message Sequence Charts (MSCs) and BPEL4WS. In [4-6], the authors established a conversation oriented framework to specify web service compositions and reason about their global behaviors. In [7], the authors use CTR-S (Concurrent Transaction Logic) to specify the contracts in semantic web services, and a proof theory for CTR-S can illustrate the use of this logic for modeling and reasoning about web service contracts. In [8] the author describes an approach to modeling web services specified in the language BPEL4WS with the help of Petri nets, then further model check the correctness requirements of a process. Although above approaches can effectively verify the process of interacting of web services, they don't take into account the inner behaviors of each web service.

In [9], the authors use CCS (a kind of process algebra) to formalize WSCI, and then be in a position to check whether two or more web services are compatible to interoperate or not, and if not, whether the specification of adaptors that mediate between them can be automatically generated. Though this approach can effectively formalize WSCI, but the verification is not intuitionist.

In this paper, we will show the benefit of using automata to formalize WSCI. We verify the compatibility among web services through a client/server model. We will show that it guarantee the soundness of collaboration of web services.

3 Modeling Web Services Based on WSCI with DFA

The Web Services Choreography Interface (WSCI) defines the overall choreography describing the messages between Web services that participate in a collaborative exchange. In intuition, we can use automata to formalize WSCI because the

description of WSCI is about the flow message of exchanged by a web service. First, we show how to formalize the basic operations of web service. The type of basic operations of web service can be divided into four types as shown in the left of fig 1. Here, we only list some descriptions of WSCI, the details of WSCI can be found in [2].

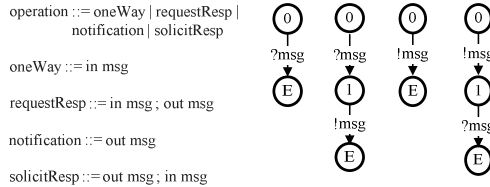


Fig. 1. Model of basic operations

In the right of fig 1, we list the model of basic operations of web service. We use the signal “?” to express “in msg”, the signal “!” to express “out msg”, each circle express one state of the web service: “O” means the start state, “E” means the end state and other means the intermediate state.

Then, we can build the flow model of web service based on the model of basic operations. This, we only deal with the four classic structures of flow model: sequence, concurrent, choice and loop. We think the four structures are enough to express the flow model of interacting web services.

Structure	WSCI sample	Model	Structure	WSCI sample	Model
SEQUENCE	<pre> <sequence> <action 1> <action 2> </sequence> </pre>		CHOICE	<pre> <all> <action 1> <action 2> </all> </pre>	
CONCURRENT	<pre> <switch> <case> <condition> <condition1> </switch> <action 1> <action 2> <default> </switch> </pre>		LOOP	<pre> <while> <action 1> </while> </pre>	

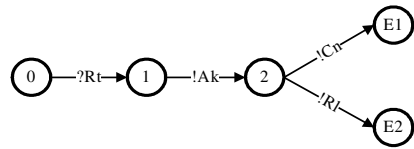


Fig. 2. The model of four structures and an example of booking web service

In the left of fig 2, we display the model of four kinds of structure. We use signal “e” to express the empty transition. In concurrent model, we use the signal “x” to express the concurrent result, “x” means the result of concurrent actions can be simply constructed from Cartesian product of all its branches. And now, based on this model, we can get a model of the whole web service. The original model of the service is a NFA, and we can turn it into a DFA.

The right of fig 2 is a model of a web service of booking a ticket. It expects to receive a message *BookingRequest* (Rt) from any clients first, then replies a message *BookingAck* (Ak), Finally, it sends a message *Confirm* (Cn) or *Refusal* (RI) depending on judgment.

Definition 1. The model of a web service is $A=(\Sigma_{inp}, \Sigma_{out}, \Delta, T, s_0, F)$. In A , Σ_{in} is the set of all input messages, Σ_{out} is the set of all output messages. T is a finite set of states, s_0 is the initial state, F is the set of final states, and Δ is the transition function: $T \times (\Sigma_{in} \cup \Sigma_{out}) \rightarrow T$.

Definition 2. The local conversation set of a web service is L consisted of the set of all passed messages when the web service executes from the initial state to the final states.

Definition 3. The marked conversation set of a web service is M based on L , it marks each conversation of L with “0” to express the “receiving message” and “1” to express the “sending message”.

Definition 4. A successful execution of a web service means that certain one of its local conversations can be completely realized during this execution.

4 Verifying Web Services Collaboration

Once we have translated web services written in WSCI into a formal model--automata, we can check whether two or more web services are compatible in collaboration or not. This, we divided the question into two parts according to the number of web services in collaboration: two web services and more web services.

4.1 Two Web Services Cases

Definition 5. (The role of a web service): R express the role of a web service in collaboration with another web service, we divide R of a web service into three types: *client*, *server* and *both*. If the initial state of one web service is only staying in the state of receiving message, then the web service is a *server*, if the initial state is only staying in the state of sending message, then the web service is a *client*, otherwise the web service is *both*. If the role a web service is *both*, it means that the web service may be a *client* or a *server* in any collaboration.

The conversation classes based on the role of one web service are consisted of two sets: S_C and S_S . S_C expresses the conversation set based on the role of client and S_S expresses the conversation set based on the role of server.

The previous example shows that $S_C=\Phi$ and $S_S=\{RtAkCn, RtAkRl\}$. And according to definition 3, the corresponding $M_C=\Phi$ and $M_S=\{011, 011\}$.

Definition 6. (Compatibility of Two Web Services): let A and A' are two arbitrary web services, A is the client and A' is the server, in any collaboration with them, if A' has a successful execution, A must have a successful execution, and the conversations generated from their executions must be identical, we think they are compatible.

Theorem 1. let A, A' are models of two web services, $A=(\Sigma_{inp}, \Sigma_{out}, \Delta, T, s_0, F)$ and $A'=(\Sigma'_{inp}, \Sigma'_{out}, \Delta', T', s_0', F')$, A is the client and A' is the server, A and A' are compatible in a collaboration iff:

1. $S_{S_{A'}} \subseteq S_{C_A}$

2. For each conversation $l_i \in S_{SA}$, its corresponding marked conversation should satisfy: $\delta(M_{CA}(l_i))=M_{SA}(l_i)$, δ means a function: $0 \rightarrow 1$ and $1 \rightarrow 0$.

The proof of this theorem is omitted here.

Example 1. A client is shown in fig 3, from its model, we know it is not compatible with the model of fig 2, because the conversation “RtAkRI” can not appear in the execution of the client.

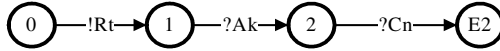


Fig. 3. A model of a client

4.2 More Web Services Cases

When the number of web services in a collaboration is more than two, the analysis of compatibility is different to the above, we can't simply apply the client/server model. First, we give the concept of the global conversations.

Definition 7. (Global Conversation): A global conversation is a flow of messages interaction among collaborating web services. We express it with G .

Definition 8. (Compatibility of More Web Services): If there exists collaboration among $n(n \geq 3)$ of web services, and for the given global conversation G , each web service can have a successful execution, then these n web services are considered compatible in G .

For the collaboration among n of web services, it's difficult to use simple Client/Server model to solve the complicated problems. Thus we need to make some changes. The following shows the **judgment algorithm**:

```

For i=1 to n
  /*First judge the role of service Ni. During the
  global conversation, if the first message of Ni
  invoked is an output message, then the service Ni
  will be Client E1 and other n-1 web services will
  be Server E2. Otherwise Ni will be Server E2 and
  other n-1 web services will be Client E1.
  Then judge whether Ni is compatible with other
  n-1 web services in the following steps.*/
  
```

1. Generate an automata model V to describe the global conversation G .
2. For the model V , assume there is j out of $n-1$ web services related to service Ni . Then P is the projection of V on the j web services.
3. Compute M_p . For each message in P , if it's an output message from Ni then set flag '0'; otherwise it's an input message of Ni then set flag '1'.
4. If Ni is the client then those $n-1$ services will be the server and $P=S_{SE2}$. Otherwise those $n-1$ ones will be the server and $\bar{P}=S_{CE1}$.

5. By Theorem 1, determine whether service N_i is compatible with other $n-1$ ones. If compatible, go on with service N_{i+1} . Otherwise complete the algorithm and exit.

6. When all the services have been judged, we can conclude they are compatible.

End

5 Summary and Future Work

In this paper, we described an approach to model web services specified in WSCI with automata. Based on this model, we can effectively check whether two or more web services are compatible in collaboration or not. In future, we are going to build up an integrated modeling, verifying, and testing environment for web services.

References

- [1] BPEL4WS Specification: Version 1.1 2003 <http://www-106.ibm.com/developerworks/library/ws-bpel>
- [2] W3C, "Web Service Choreography Interface (WSCI) 1.0," World Wide Web Consortium (2002), available at <http://www.w3.org/TR/wsci>.
- [3] H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-based verification of web service compositions. In Proc. 18th IEEE Int. Conf. on Automated Software Engineering (ASE), 2003.
- [4] X. Fu, T. Bultan, and J. Su. Conversation protocols: A formalism for specification and verification of reactive electronic services. In Proc. Int. Conf. on Implementation and Application of Automata (CIAA), 2003.
- [5] T. Bultan, X. Fu, R. Hull, and J. Su. Conversation specification: A new approach to design and analysis of e-service composition. In Proc. Int. World Wide Web Conf. (WWW), May 2003.
- [6] X. Fu, T. Bultan, and J. Su. Analysis of interacting BPEL Web Services. Proceedings of 13th International Conference on World Wide Web (WWW), 2004.
- [7] H. Davulcu, M. Kifer and I.V. Ramakrishnan. CTR-S: A Logic for Specifying Contracts in Semantic Web Services. In Proceedings of 13th International Conference on World Wide Web (WWW), 2004.
- [8] Holger Schlingloff. Modeling and Model Checking Web Services. Electronic Notes in Theoretical Computer Science 126 (2005) 3–26
- [9] Antonio Brogi, Carlos Canal, Ernesto Pimentel, Antonio Vallecillo. Formalizing Web Service Choreographies. Electronic Notes in Theoretical Computer Science 105 (2004) 73–94.

A Novel Resource Description Based Approach for Clustering Peers

Xing Zhu, Dingyi Han, Yong Yu, and Weibin Zhu

Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai, China
{redstar, tony, dany, yyu}@apex.sjtu.edu.cn

Abstract. Clustering similar peers could benefit document retrieval in P2P systems. In this paper, we suggested a resource description based approach to cluster peers according to their topics. By combining the *topic model* (an extension of *language model*) technique and *fuzzy set theory*, we solve the key problems about resource generation and peer similarity calculation. Experiments performed on the standard data sets prove that our approach is more effective than the traditional method.

1 Introduction

File sharing is an important application of Peer-to-Peer (P2P) Systems. With more and more text documents being shared, there ought to be some effective information retrieval mechanisms in these text-document-sharing P2P applications.

We call that the documents of the same category, i.e. being about the same *topic*, *similar documents*. The document collections which are made up of similar documents are called as *similar collections* and the peers which own similar collections are called as *similar peers*¹. Intuitively, clustering similar peers will help to discover useful resources and prune the searching space. Therefore, clustering similar peers will benefit information retrieval in P2P systems.

In this paper, we suggest an approach to cluster similar peers based on the *resource description*[1] of the document collection in a peer. The similarity between two peers is determined by the similarity between their resource description.

Similar to the statistical resource description generation approach, we use a weighted term list to describe the document collection. The basic technique to determine the term weight is based on *Topic Model*[2], an extension of *Language Model*[3]. After building the resource description, we borrow the *fuzzy correlation coefficient* from *fuzzy clustering* [4] to deduce the similarities among different peers.

¹ In order to simplify our discussion, we just consider the situation where all the documents in a peer have one *topic*. In fact, the situation in which the document collections with multiple *topics* is a extension of the basic *one-topic* situation. Hence, we can transfer the approach that is workable in *one-topic* situation to *multi-topic* situation.

We organize our paper as follows. In Section 2, we describe our mechanism. The experiments and experimental results are discussed in Section 3. We conclude the whole paper in Section 4.

2 Our Resource Description Based Approach

2.1 Background

In essence, Language model suggests a new method to describe the relations between terms and documents, i.e., we can consider that a term t is generated by the model of the document d in which it appears. As showed in Formula (1), the generation relationship is defined as a conditional probability $p(t|M_d)$ [3] :

$$\hat{p}(t|M_d) = \begin{cases} \frac{p_{ml}(t, d)^{1-\hat{R}_{t,d}} \times p_{avg}(t)^{\hat{R}_{t,d}}}{|C|_t} & \text{if } tf(t, d) > 0 \\ \text{otherwise} & \end{cases} \quad (1)$$

If all the documents in a collection are about the same topic T , we can consider that the selected terms in the resource description are generated by a unified model of T , i.e. the *Topic Model* (M_T) [2]. Suppose that d_j is a document from the document collection C with topic T . Because d_j is about T , we consider that $P(t_i|M_{d_j})$ is a sample value of $P(t_i|M_T)$. Therefore, we estimate $P(t_i|M_T)$ according to the following formula:

$$\hat{p}_{ev}(t_i|M_T) = \frac{\sum_{d_j \in C} p(t_i|M_{d_j})}{|C|_d} \quad (2)$$

where $p(t_i|M_{d_j})$ could be estimated according to Formula (1); $|C|_d$ is the number of documents in C with T . In fact, we use the *expected value* of the samples of $p(t_i|M_T)$ as its estimate.

According to [2], there is another way to estimate the probability:

$$\hat{p}_{ml}(t_i|M_T) = \frac{1}{|C|_d} \times \sum_{t_i \in d_j} \frac{tf(t_i, d_j) + 0.01}{|d_j|_t + 0.01n} \quad (3)$$

where n is the vocabulary size of d_j . In essence, this method use the maximum likelihood estate as the estate of $p(t_i|M_T)$.

2.2 Generate the Resource Description

Term Selection. According to our observation, the words that are important to the topic will have higher document frequency. Hence, We select the words as the terms to construct the resource description according to following rules:

$$df(t_i, C) \geq \text{DFT} \quad (4)$$

where $df(t_i, C)$ is the document frequency of term t_i and DFT is a predefined threshold. Fig.1 shows an example where $\text{DFT} = 2$. In this example, word t_2 only appear in one document d_2 , so it is filtered out from the term list.

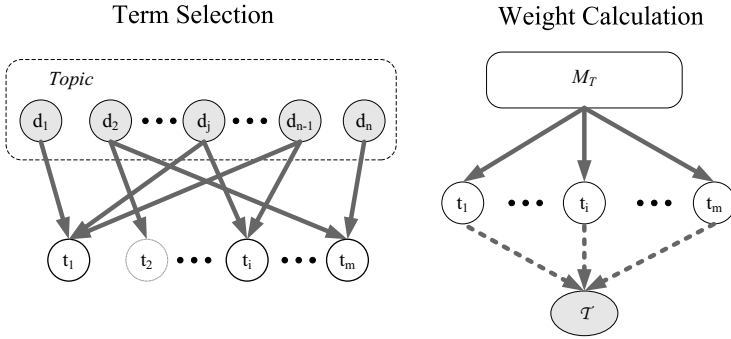


Fig. 1. The Illustration of Term Selection and Weight Calculation

Weight Calculation. In fact, the relationship between a term and a topic is a typical *fuzzy relation*. In our implement, we take the topic of a document collection C as a fuzzy set \mathcal{T} and the terms in C as the elements of \mathcal{T} . The membership function ($\mathcal{F}(t_i, \mathcal{T}) \rightarrow [0, 1]$) defines the weight of term

$$w_i \leftarrow \mathcal{F}(t_i, \mathcal{T}) = \frac{p(t_i|M_T)}{\max_{t_j \in C}(p(t_j|M_T))} \tag{5}$$

where w_i is the weight of term t_i ; $p(t_i|M_T)$ could be estimated by $\hat{p}_{ev}(t_i|M_T)$ or $\hat{p}_{ml}(t_i|M_T)$; and $t_j \in C$ means that term t_j appears in at least one document of C . If $t_i \notin C$, $p(t_i|M_T) = 0 \Rightarrow w_i = 0$.

As showed in Fig.1, *topic model* is the basic technique in weight calculation.

2.3 Calculate the Similarity

Just as what has been mentioned, we map the topic of a document collection to a fuzzy set \mathcal{T} . Hence, we can deduce the similarity between two peers $peer_i$ and $peer_j$ by using the *fuzzy correlation coefficient* between two fuzzy variables \mathcal{T}_i and \mathcal{T}_j

$$sim(peer_i, peer_j) \leftarrow \tilde{\rho}(\mathcal{T}_i, \mathcal{T}_j) = \frac{\sum_{k=1}^{m'} |w_{ik} - \bar{w}_i| \cdot |w_{jk} - \bar{w}_j|}{\sqrt{\sum_{k=1}^{m'} (w_{ik} - \bar{w}_i)^2} \cdot \sqrt{\sum_{k=1}^{m'} (w_{jk} - \bar{w}_j)^2}} \tag{6}$$

, where w_{ik} is the weight of term t_k from $peer_i$; \bar{w}_i is the average weight of all the terms from $peer_i$. m' is the cardinality of UT , which is a union term set of TS_i (term set of the resource description of peer i) and TS_j (term set of the resource description of peer j). For term $t_m \in (UT - TS_i)$, $w_{im} = 0$. Analogously, for term $t_n \in (UT - TS_j)$, $w_{jn} = 0$.

By adopting the above similarity calculating scheme, finding similar peers and grouping them into peer clusters is a *fuzzy clustering*[4] process. To our best knowledge, it is our innovation to judge the similarities among peers by combining fuzzy sets theory and language model techniques.

Table 1. Information about the Test Beds

Test bed	Topics in the Test bed
<i>RCV1-1</i>	CCAT; ECAT; GCAT; MCAT
<i>RCV1-2</i>	E13, E131; E14, E141, E142, E143; E21, E211, E212; E31, E311, E312, E313; E41, E411; E51, E511, E512; E61; E71
<i>RCV1-3</i>	C11, C15, C151, C31; E11, E21, E211, E212; G15, G151, G152, G153, GCRIM, GDEF; M11, M13, M14, M131, M132, M14, M141, M142, M143
<i>TREC5</i>	Science and Technology; Medical and Biological; Law and Government; International Relations; International Politics; International Finance; International Economics; Environment

3 Experiments and Evaluation

3.1 Data Sets

We performed our experiments in two data sets: Reuters Corpus Volume 1 (RCV1) [5] and TREC 5 (disk 5) [6]. In RCV1, the topic information was directly used as the topic of the peer in our experiments. In TREC5, we used the *domain* information in the routine task as the topics.

3.2 Experiment Design

Four test beds were constructed by using the two data sets. Table 1 lists the information of the four test beds. For each test bed, we have simulated a file sharing P2P system.

We group *similar peers* into *peer clusters* with *K-means clustering algorithm*[7]. We use *FMeasure*[7] to measure the cluster quality:

$$F(T_i, C_j) = \frac{2}{\frac{1}{R(T_i, C_j)} + \frac{1}{P(T_i, C_j)}}$$

The higher the cluster quality is, the closer is $F(T_i, C_j)$ to 1.

In our experiments, we evaluate the effectiveness of our resource description *generation scheme* in clustering similar peers. We try to clarify the influences of term selection scheme and the weight calculation method on peer clustering. We also compare the effectiveness of our approach (*topic model*-based approach, *tm approach*) with that of the traditional approach directly based statistical information (*term frequency*-based approach, *tf approach*) in peer clustering.

3.3 Experimental Data

In our implement of K-means algorithm, resource descriptions (weighted term lists) are taken as the vectors; the weights of the terms are the corresponding

Table 2. The Influences of DFT and $P(t_i|M_T)$ Estimation Method

		<i>RCV1-1</i>	<i>RCV1-2</i>	<i>RCV1-3</i>	<i>TREC5</i>
DF 2	$\hat{p}_{ev}(t M_T)$	1	0.979	0.952	1
	$\hat{p}_{mi}(t M_T)$	1	0.979	0.948	1
DF 4	$\hat{p}_{ev}(t M_T)$	1	0.978	0.944	0.999
	$\hat{p}_{mi}(t M_T)$	1	0.975	0.942	0.999
DF 6	$\hat{p}_{ev}(t M_T)$	1	0.974	0.941	0.999
	$\hat{p}_{mi}(t M_T)$	1	0.971	0.939	0.998
DF 8	$\hat{p}_{ev}(t M_T)$	1	0.966	0.944	0.997
	$\hat{p}_{mi}(t M_T)$	1	0.964	0.944	0.995
DF 10	$\hat{p}_{ev}(t M_T)$	0.991	0.962	0.938	0.989
	$\hat{p}_{mi}(t M_T)$	0.991	0.961	0.937	0.987

Table 3. Comparing the effectiveness of different approaches (*FMeasure*)

	<i>RCV1-1</i>	<i>RCV1-2</i>	<i>RCV1-3</i>	<i>TREC5</i>
<i>tf Approach</i>	1	0.965	0.931	0.733
<i>tm Approach</i>	1	0.974	0.941	0.999

coordinates in the vector. All the vectors are normalized. In *tm approach*, the distance function in K-means algorithm is Formula(6). In *tf approach*, $\cos(\mathbf{v}_i, \mathbf{v}_j)$ is used as the distance function.

Evaluation of Resource Description Generation Scheme. Table 2 shows the influence of the different DFTs and different $P(t_i|M_T)$ estimation methods on the accuracy of the clustered results. *FMeasure* in each test bed is the average value of the *FMeasures* of the topics in the corresponding test bed. **DF 2** means $DFT = 2$. $\hat{p}_{ev}(t|M_T)$ means that when calculating the term weight, $\hat{p}_{ev}(t|M_T)$ is used as the estimate of $P(t_i|M_T)$.

In this table we can see that *FMeasures* decrease slightly with the increasing of DFT. The maximum dropping rate is around 1.7%. In every testbed, $\hat{p}_{ev}(t|M_T)$ slightly surpass or equals to $\hat{p}_{mi}(t|M_T)$ in finding similar peers.

Comparing with the Approach Based on the Statistical Method. Traditionally, $tf(t_i, C)$, the total term frequency of t_i in a document collection C , is adopted to select terms. Words were sorted according to their $tf(t_i, C)$ in the document collection and the top n words were selected to construct the term list. The weight of the selected term is $\log(tf(t_i, C) + 1)$. The value of n will affect the accuracy in clustering. According to [8], what we found is that when $n = 2000 \sim 4000$, there was no significant difference in the clustering results. When $n \leq 1000$, the performance degraded obviously.

We compared the performance of the *tm approach* with the *tf approach* in finding similar peers. In the *tm approach*, the standard to select term is $DFT = 6$. The standard in *tf*-based approach is $n = 2500$. Table 3 shows the results of the

comparative experiments. Our mechanism performed better in *RCV1-2*, *RCV1-3* and *TREC5*. Especially in *TREC5*, our mechanism significantly surpassed the *tf approach*.

4 Conclusions and Future Work

In this paper, we suggest a resource description-based mechanism to clustering *similar peers* in P2P systems. We solved the problems about resource generation and peer similarity calculation. One contribution of our work is that we manage to get more accurate term weight by combining the *topic model* technique and *fuzzy set theory* in resource generation. The other contribution is that we infer the similarity between peers by the *fuzzy correlation coefficients* between *fuzzy variables*. Our experiments proved that our approach is more effective in clustering peers than the traditional approach which directly utilizes the term statistical information.

References

1. Meng, W., Yu, C., Liu, K.L.: Building efficient and effective metasearch engines. *ACM Computing Surveys* **34** (2002) 48–89
2. Xu, J., Croft, W.B.: Cluster-based language models for distributed retrieval. In: *Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkley, CA USA, ACM SIGIR (1999) 254–261
3. Ponte, J.M., Croft, W.B.: A language modeling approaching to information retrieval. In: *Proceedings of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM SIGIR (1998) 275–281
4. Yang, M.S.: A survey of fuzzy clustering. *Math. Comp. Modelling* **18** (1993) 1–16
5. Rose, T., Stevenson, M., Whitehead, M.: The reuters corpus volume 1 - from yesterday's news to tomorrow's language resources. In: *Proceedings of the 3rd International Conference on Language Resources and Evaluation*. (2002) 29–31
6. Voorhees, E.M., Harman, D., eds. In Voorhees, E.M., Harman, D., eds.: *The Fifth Text REtrieval Conference (TREC-5)*. (1996)
7. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. Technical Report #00-034, Department of Computer Science and Engineering, University of Minnesota (2000)
8. Gauch, S., Wang, J., Rachakonda, S.M.: A corpus analysis approach for automatic query expansion and its extention to multiple databases. *ACM Transaction on Information System* **17** (1999) 250–269

The Segmentation of News Video into Story Units

Liu Huayong¹ and Zhang Hui²

¹ Department of Computer Science, Central China Normal University,
Wuhan 430079, Hubei, PR China
hyliuwuhee@hotmail.com

² Finance Department of Business School, Wuhan University,
Wuhan 430072, Hubei, PR China
zhanghui994@sohu.com

Abstract. The research proposes an approach of story segmentation for news video using multimodal analysis. The approach detects the topic-caption frames, and integrates them with silence clips detection, as well as shot segmentation to locate news story boundaries. On test data with 135,400 frames, the accuracy rate 85.8% and the recall rate 97.5% are obtained. The experimental results show the approach is valid and robust.

1 Introduction

Large amount of news videos are available. We need an automatic and effective tool to segment these news video into single story units. These story units are used for indexing to support further browsing and retrieval by the users. In order to characterize the video content, video structure parsing is required for indexing. Many literatures have addressed the shot boundary detection techniques, such as Ref. [1]. Some researchers have presented scene segmentation or extraction algorithms, for example, Ref. [2] presents a novel algorithm that uses number of interpolated macro blocks in B-frames to identify the sudden scene changes detection and statistical features for gradual scene change detection. Ref. [3] proposes a fast scene change detection algorithm using direct feature extraction from MPEG compressed videos. Some literatures also contain the approaches about story detection or segmentation, for instance, Ref. [4] has introduced an approach to extract story units from long programs for video browsing and navigation by time-constrained clustering of video shots and analysis of scene transition graph.

Due to development limitation of machine vision and pattern recognition, now it is still difficult to automatically extract high-level semantic structure such as scene, story for general video programs, this problem is also not solved completely in semantic content level in late research works listed above. We should exploit more clues, not just depend on the visual information analysis. For example, Ref. [5] has used image analysis techniques to automatically parsing news video. Their algorithm that locates and identifies anchorperson shots

is based on the a priori knowledge and assumes that each news item starts with an anchor shot followed by a sequence of news shots. Due to his assumption limitations, news items that start without anchorperson shots and that read by anchorperson without news shots can not be identified by their system. This paper presents an approach, which integrates visual, auditory and textual information to segment news stories. It overcomes the limitations of the algorithm mentioned above and is helpful for users to efficiently manage their video resources.

2 Story Segmentation of News Video

Fig. 1 shows our overall system components. Shot boundary detection module segments video stream into a sequence of shots. Caption text detection module identifies text event to locate the clips composed of topic caption frames. In silence clip detection module, silence segments are identified. Then audio-visual features and text information are integrated to segment news video into individual stories.

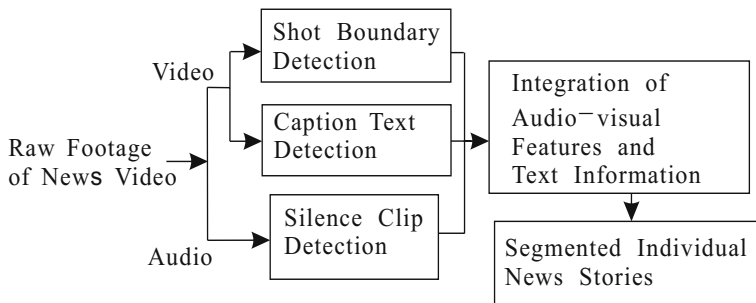


Fig. 1. Overall system components

2.1 Shot Boundary Detection

Because of spot and time limitations of news video in shoot and edition, the abrupt shots of news video obtain a rate of 90%. The gradual transitions generally appear in the head, tail of news programs. Even if they occur in the main body of news programs, they generally occur inside of the news story and do not locate at news story boundaries, so we just consider the detection of abrupt transition because there should be abrupt transition between two consecutive news stories to a great extent.

The method of X^2 histogram matching mentioned in Ref.[6] is used in this paper to measure the content change between each contiguous frame pairs. We define E_{at} as the abrupt shot transition event.

2.2 Topic Caption Text Detection

A row or several rows of captions added by late edition must appear to express the meaning of news story in the start or middle, as shown in Fig. 2(a). They generally appear in the specific position (for example, bottom-left part of TV screen) and last several seconds. These captions are defined as topic captions, and the frame containing topic captions is defined as topic-caption frame. The appearance and the disappearance of these captions are defined as text event.

In order to detect the topic-caption frames, we present a novel specific region text detection algorithm based on threshold. For L long, $N \times M$ size of video frame sequence F_t^R , $t(1, L)$ is the number of frame. $R = \{(i, j) : i(1, N), j(1, M)\}$, $R^* = \{(i, j) : i(1, N), j(0.75 \times M, M)\}$. R is the region of the whole frame, and R^* is the region defined as the 1/4 field of screen bottom where text events appear. So we can get two frame difference sequences: the whole frame difference sequence d_t^R and the part frame difference sequence $d_t^{R^*}$, and for random $t(1, L)$, $d_t^R = F_{t+1}^R - F_t^R$, $d_t^{R^*} = F_{t+1}^{R^*} - F_t^{R^*}$.

If there are shot transitions when text events occur, the algorithm is done as follows. T_{text} and T_{shot} are defined as the global threshold of text event and shot abrupt transition. If $d_t^{R^*} > T_{\text{text}}$ and $d_t^R < T_{\text{shot}}$, there is a text event between frame of t and $t + 1$. If $d_t^{R^*} > T_{\text{text}}$ and $d_t^R > T_{\text{shot}}$, there is a shot abrupt transition between frame of t and $t + 1$. If the backgrounds have changes when text events occur, the algorithm is done as follows. A slider detection window is defined using the following expression,

$$W_l(R^*) = \{d_{t-m}^{R^*}, \dots, d_{t-1}^{R^*}, d_t^{R^*}, d_{t+1}^{R^*}, \dots, d_{t+m}^{R^*}\}. \tag{1}$$

where l is the window length and $l = 2m + 1$. If $d_t^{R^*} \geq \alpha \times d_j^{R^*}$, $d_j^{R^*} \in W_l(R^*)$, $d_j^R < T_{\text{shot}}$, $d_t^R < T_{\text{shot}}$ and $j \neq t$, there is a text event between frame of t and $t + 1$. The following parameters are chosen, $\alpha = 2$, $m = 2$.

Some else captions like the dialogue captions between a reporter and an interviewee may be added in the same region, as shown in Fig. 2(b). In order to avoid wrong detection, the fixed region RF within the dotted lines(as shown in Fig. 2(c)) is chosen. The color Euclidean distances between the RGB values of RF and the standard blue and white values are calculated respectively. If the distances sum is lower than a certain defined threshold, then these frames are potential topic-caption frames. In the next step, the two horizontal edges of caption text region are detected using edge detection algorithm described in Ref.[7] to confirm the topic-caption frame.

Based on analysis above, the topic-caption clips $TC(n)$ can be obtained, and it can be expressed as $TC(n) = [V_{s_n}, V_{e_n}]$, $n = 1, 2, \dots$, and $V_{s_n} < V_{e_n}$. V_{s_n} and V_{e_n} represent the corresponding frame number of the topic captions start and end respectively. We choose the frame at the position $\text{INT}((V_{s_n} + V_{e_n})/2)$ as the topic-caption frame of the n th story and $\text{INT}(a)$ is defined to get the integrate part of a . Then the array $Ftc(n)$ of topic-caption frames sequence can be obtained, which is expressed as $Ftc(n) = \{Ftc_1, Ftc_2, \dots, Ftc_n\}$.

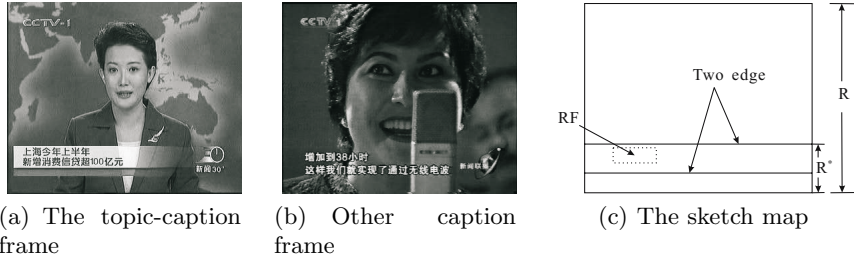


Fig. 2. The example and sketch map of topic caption detection structure

2.3 Silence Clip Detection

According to lots of experimental observation of news video, we find out that a relatively long silence clip must exist in two continuous news story boundary. The frame containing silence clip is defined as silence-clip frame. In our system, the audio stream is defined as two layers including audio frame and audio clip. An audio frame lasts about 20ms, and continuous 30 audio frames are chosen as an audio clip.

Both energy and ZCR measures are used to detect silence clips. If the short-time energy function value of an audio frame is lower than the defined threshold T_e and its short-time zero-crossing rate is lower than another defined threshold T_{zcr} , then the audio frame is indexed as a silence frame SF , also define $SF^S = 1$. For audio clip $AC_{V_{t1}, V_{t2}}^i$, V_{t1} and V_{t2} represent the start and the end frame number of AC^i respectively, $AC_{V_{t1}, V_{t2}}^{iS} = \sum_{V_t=V_{t1}}^{V_{t2}} SF_{V_t}^S$, if $AC_{V_{t1}, V_{t2}}^{iS} \geq \beta$, then $AC_{V_{t1}, V_{t2}}^i$ is defined as the i th silence clip, and $Esc_{V_{t1}, V_{t2}}^i$ is defined as the silence event. The following parameters are chosen, $T_e = 10000$, $T_{zcr} = 0.02$, $\beta = 24$. Based on analysis above, we can get a series of silence clips $SC(n) = \{SC_{V_{s1}, V_{e1}}^1, SC_{V_{s2}, V_{e2}}^2, \dots, SC_{V_{sn}, V_{en}}^n\}$, $n = 1, 2, \dots$, and $V_{sn} < V_{en}$. V_{sn} and V_{en} represent the corresponding start and end frame number of SC^n respectively.

2.4 News Stories Segmentation Algorithm

For a news story NS_k , it has only one topic-caption frame Ftc_k , and there should be one or more silence clips before and after appearance of Ftc_k in temporal axis. These silence-clip frames should have abrupt transition except that a whole news story is inside of an anchorperson shot and it has not live broadcasts, so we can detect shot transition of these specific silence-clip frames between two consecutive topic-caption frames to locate the story boundaries.

The algorithm is described as follows: 1. For $SC_{V_{sn}, V_{en}}^n$, if $[V_{sn}, V_{en}] \in [Ftc_k, Ftc_{k+1}]$, $Esc_{V_{sn}, V_{en}}^n \cap E_{at} \neq \Phi$, then the frame at the position $INT((V_{sn} + V_{en})/2)$ is chosen as the story boundary between NS_k and NS_{k+1} . 2. For $SC_{V_{si}, V_{ei}}^i$ and $SC_{V_{sj}, V_{ej}}^j$, if $[V_{si}, V_{ei}] \in [Ftc_{k-1}, Ftc_k]$, $[V_{sj}, V_{ej}] \in [Ftc_k, Ftc_{k+1}]$, $Esc_{V_{si}, V_{ei}}^i \cap E_{at} \neq \Phi$, $Esc_{V_{sj}, V_{ej}}^j \cap E_{at} \neq \Phi$, it shows that the news story NS_k is inside of one

Table 1. Results for the visual streams analysis

News Videos	Actual number	shot	Actual boundaries	story	Output of boundaries	Missing	False
	A		B		C	D	E
News1	295		28		35	1	8
News2	301		29		34	1	6
News3	256		24		28	3	7
Total	852		81		97	5	21

anchorperson shot and there is no abrupt transition around news story NS_k . We calculate $\lambda = V_{s_\lambda} - V_{e_\lambda} = \max\{V_{s_i} - V_{e_i}\}$, $[V_{s_\lambda}, V_{e_\lambda}] \in [Ftc_{k-1}, Ftc_k]$, and choose the frame at the position $\text{INT}((V_{s_\lambda} + V_{e_\lambda})/2)$ as the story boundary between NS_{k-1} and NS_k . Then calculate $\Theta = V_{s_\Theta} - V_{e_\Theta} = \max\{V_{s_j} - V_{e_j}\}$, $[V_{s_\Theta}, V_{e_\Theta}] \in [Ftc_k, Ftc_{k+1}]$, and choose the frame at the position $\text{INT}((V_{s_\Theta} + V_{e_\Theta})/2)$ as the story boundary between NS_k and NS_{k+1} .

3 Experimental Results and Evaluation

The material is three days of MPEG-1 CCTV news with frame dimension of 352×288 pixels and frame rate of 25 frames per second, which is selected from our video database randomly. The test data set lasts one and a half hours or so in total and contains 135,400 frames. In order to test the validity of our story segmentation algorithm, the test data are labelled manually as a standard contrast.

Table 1 shows the results only using visual stream analysis algorithm (including shot boundary and topic caption text detection). We can calculate the accuracy rate $P = 1 - E/C = 1 - 21/97 = 78.3\%$ and the recall rate $R = 1 - D/B = 1 - 5/81 = 93.8\%$. This results show that only using visual streams analysis is not sufficient. Table 2 shows the results for the multimodal analysis. The precision is improved to 85.8%, and the recall value to 97.5%. This results show that multimodal analysis method is effective and robust.

The experiment also shows for the news that only containing anchorperson shots the algorithm is effective. For example, text starts at frame 3091 and text ends at frame 3198, and there is no abrupt transition around them. The silence-clip frames are located at around frame 2798 and frame 3450, and then the story boundaries are located at frame 2798 and frame 3450 respectively, which is same to the manually labelled results. Missing boundaries are mainly caused by disturbs of abrupt transitions occurring on the topic-caption frames, and text event is considered as general shot transition. False boundaries are mainly caused by disturbs of other captions such as the captions of interviewee personal introduction or the dialogue between the reporter and the interviewee.

Table 2. Results for the multimodal analysis

News Videos	Actual number A	shot B	Actual story boundaries C	Output of story boundaries D	Missing E	False
News1	295	28	33	0	5	
News2	301	29	32	1	4	
News3	256	24	27	1	4	
Total	852	81	92	2	13	

4 Conclusions

Extracting high-level semantic content from video flows such as news video is hot spot in the researches of video database. This paper presents an effective approach of story segmentation for news video from the point of view of the integration of image, auditory and textual cues. Though the approach is designed for parsing TV news, its analysis of text event detection, as well as the integration strategy of audio-visual cues, can also be applied to the scene segmentation and video retrieval of other video types in future work.

Acknowledgements. The research was supported by the Hubei Nature Science Fund in China under Grant No. 2005ABA246.

References

1. Wei Jyh Heng, King N Ngan: Post Shot Boundary Detection Technique: Flashlight Scene Determination. Fifth International Symposium on Signal Processing and its Applications, ISSPA'99 (1999) **1**: 447–450
2. Fernando W A C, Canagarajah C N, Bull D R: Scene Change Detection Algorithms for Content-Based Video Indexing and Retrieval. IEEE Electronics & Communications Engineering Journal (2001) **3**(3): 117–126
3. Lee Seong Whan, Kim Young Min, Choi Sung Woo: Fast Scene Change Detection Using Direct Feature Extraction from MPEG Compressed Videos. IEEE Transactions on Multimedia (2000) **2**(4): 240–254
4. Yeung Minerva, Yeo Boon Lock, Liu Bede: Extracting Story Units from Long Programs for Video Browsing and Navigation. IEEE Proc. of Multimedia'96 (1996) 296–305
5. Zhang H J, Tan S Y, Smoliar S W, et al: Automatic Parsing and Indexing of News Video. Multimedia Systems (1995) (2): 256–266
6. Nagasaka A, Tanaka Y: Automatic Video Indexing and Full-Video Search for Object Appearance. Second Working Conf. on Visual Database Systems, North-Holland: Elsevier Science Publ BV (1991)
7. Zabih Ramin, Miller Justin, Mai Kevin: Feature-Based Algorithms for Detecting and Classifying Scene Breaks. Fourth ACM Conf. on Multimedia, ACM Press (1995) 189–200

Process Controlling and Monitoring Scheme for Distributed Systems with Fault-Tolerance by Using Web Services

YunHee Kang and KyungWoo Kang

Department of Computer and Communication Engineering,
Cheonan University,
115, Anseo-dong,
Cheonan 330-704, Choongnam, Republic of Korea
{yhkang, kwkang}@cheonan.ac.kr

Abstract. This paper discusses a management of controlling and monitoring of processes in a distributed system based on Linux and UNIX operating system by using Web Services. We propose a system architecture for handling processes located dispersedly in the Internet. In the design of the system architecture, the level of detail is focused on fault-tolerance aspect as part of non-functional attributes. We also show an experimental design of failure handling service named *Hydra*, which is based on Web Services. A prototype allows Linux and UNIX systems to provide more availability by restarting the process has been failure. But even more importantly, it enables us to be used by applications no modification to application structure.

1 Introduction

Management by using Web Services refers to leveraging Web Services for managing heterogeneous and distributed systems on account of their ability of reducing heterogeneity through standard interaction paradigms with XML technology [1,2]. This paper is about using Web Services as a tool for controlling and monitoring processes, and not about management of Web Services.

We introduce that Web Services provide a basis for efficient and component-based design of failure detection service. Here, we focus on management through Web Services for monitoring and controlling processes, which are running on Linux and UNIX operating system, in a distributed system. Failures are inevitable in such a distributed system built over the Internet.

In this work, we are concerned with fault-tolerance of a distributed system. We propose a system architecture for controlling and monitoring processes located dispersedly in the Internet, where the architecture level design considering fault-tolerance aspect is mainly focused. We also show an experimental design of failure handling service, which is based on Web Services. Consequently, the designed prototype system hereafter referred to as *Hydra* provides Linux and UNIX systems with more availability by restarting the process has been failure.

But even more importantly, it enables us to be used by applications no intrusion to any application structure.

This paper is organized as follows: In section 2 we describe related works including Web Services and fault-tolerance in more detail. We describe preliminaries for design of a failure detection service supporting fault-tolerance in Section 3. Section 4 shows an operational scheme we use in this paper and present its results. Finally, in Section 5, we make concluding remarks and discuss some directions for future work.

2 Related Works

2.1 Web Services

Web Services represent the next generation of enterprise infrastructure for the exchange of information and services across the Internet. W3C defines a web service that is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by XML artifacts. Web Services also support direct interactions with other software applications using XML based messages via HTTP like internet-based protocol [3,2]. The service, described in Web Services Description Language(WSDL), receives a request, processes it, and returns a response. Simple Object Access Protocol(SOAP), a lightweight protocol designed for the exchange of information, provides an envelope that encapsulates XML data for transfer through the web infrastructure [1,2].

The goal of web services is to provide high interoperability in distributed systems. In a typical web services scenario, a business application sends a request to a service at a given URI using the SOAP protocol over HTTP. Focused on distributed, decentralized environments, it provides a framework to invoke services across the Internet.

2.2 Failure Detector

Failure management is essential for building a reliable distributed system. In order to detect and diagnose failed processes, failure detection can be used as the building block to simplify the implementation [4]. Hence fault-tolerance based on failure detection is one of the means available to increase availability of delivered computational service. A failure detector(FD) is a basic element which plays a role as part of a process monitor in a distributed system [5]. Failure detectors are used in a wide variety of settings, such as network communication protocols, computer clustering management, group membership protocols, etc. A failure detector provides some information on which processes have crashed.

3 Design of a Failure Detection Service

This section describes the design and architecture of a process monitor and its status aggregation scheme based on heart-beat message for the failure detection.

3.1 Architectural Review and System Design

To help creating a framework for an architectural approach, we list some of the requirements our works come into. The first requirement is reliability in terms of service continuity. Our solution has to incorporate members that will guarantee service continuity even in the presence of failures. The architecture must consider for single point failure inherited in the component's operation and its environment. By considering fault-tolerance, the architecture covers failure detection, failure handling and recovery. To achieve the requirement we consider replication mechanism for recovery tactics. The second requirement is the capability to work in a heterogeneous environment. Basically the designed architecture must be considered highly interoperability with message level. Our solution has to be implemented in such a way as to be architecture-neutral and portable. In this perspective the client has to view the system as a virtual entity, the implementation details of which has to be completely hidden.

To take over single point of failure of *Hydra* which has the single server, we design the notification structure linked logically to detect one of servers which was crashed and to eliminate single-point failure from a group of replica.

We describe the architecture of *Hydra* in Figure 1. The structure of Hydra consists of three major functional parts: resource monitor, aggregation manager and facilitator.

Resource Monitor. The resource monitor plays a significant failure detector that watchdogs the status of processes and information about updated system load. Each process has been monitored by its resource monitor. It provides the information to an aggregation manager by a heart-beat message.

Aggregation Manager. The aggregation manager maintains the resource information which was received from the resource monitor. The resource information is delivered through an aggregating service. Its aggregating service provides a message level interface to the information provider. If a user queries the status of a specific process, the information provider sends a request to an information provider, where it stores status of processes in the persistence storage.

Facilitator. The facilitator has a function for recovering a crashed process and propagates the resources information which was provided from an information provider to other information providers. When there is no information about its status, the facilitator forwards the query to one of the information providers, which is related with the process.

In the following we describe the assumptions of an operational scheme to preserve reliable computing in the proposed architecture.

1. On a process creation, the resource monitor registers the process' physical information such as capabilities to an information provider.
2. Each information provider maintains most recently information about the process which was registered, by using periodic monitoring messages between an information provider and the process. The information provider

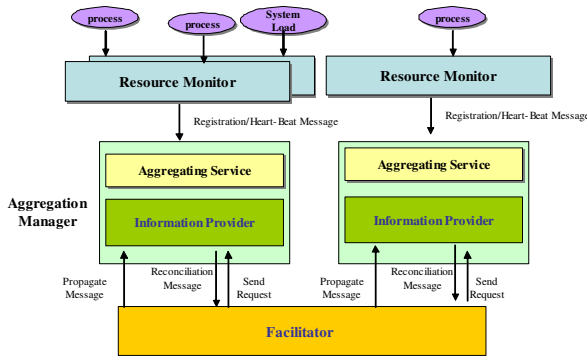


Fig. 1. The basic architecture of *Hydra*

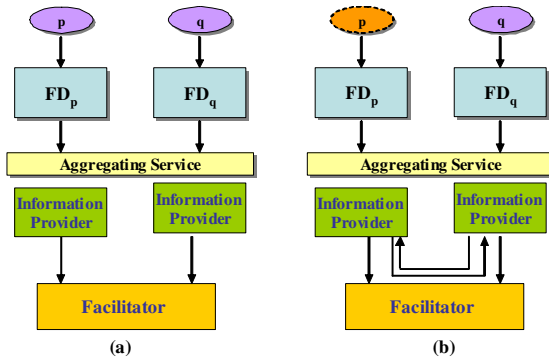


Fig. 2. The failure detection scheme

also invalidates process information such as unavailable process due to host breakdown.

3. Each information provider sends the process information gathered by the previous step (2) to a facilitator.
4. Initially one facilitator asynchronously reconciliates information about the processes with other information providers.

3.2 Basic Failure Detection Scheme

We consider a simple failure detection system of two entities FD and p which are connected through a communication link. FD represents a failure detector and p represents a process. To illustrate our point on how fault-tolerance service may be integrated with Web Services, we design the aggregating service using Web Services for aggregating information about processes based on a heart-beat message, as shown in Figure 2: Hereafter, we define an aggregating system which is composed of a failure detector, an aggregating service and information provider including an aggregating server with persistent storage.

In the first phase a process p registers itself to the information provider via resource monitor which is represented by FD_p in Figure 2, the registered process is required to send a message for advertisement. After registering the process which is monitored, the information provider propagates the status of the process to other information providers in Figure 2(a). The process p sends a heartbeat message to the failure detector FD_p ; when FD_p receives a heartbeat message, it trusts p and starts a timer with a fixed timeout value. The heart-beat message can be instantiated over SOAP. The information about a process status is delivered to the information provider into which the process is registered. When any failure has occurred in the process, this process status is delivered to an information provider in Figure 2(b). That information is to be propagated into other information providers.

4 Prototype Implementation

This section describes a prototype implementation of *Hydra* mentioned in previous sections. We mainly explain the structure for an aggregation manager. The Web Services of the prototype are implemented by Apache Axis which is a toolkit to create and consume Web Services for the Java Platform.

Figure 3 shows the system structure of the *Hydra* which consists of *aggregation manager*, *recover manager*, *failure detector*. The *aggregation manager* consists of two Web Services, *aggregating service* and *information provider*.

The *aggregating manager* maintains up-to-date information about a process which is registered to the *recovery manager* as part of the facilitator. A *process monitor*, which is as part of the *failure detector*, is written in Java. The *process monitor* watchdogs a process' state which consists of running, terminated and unknown. The *failure detector* receives information about process status from the *process monitor* watchdoging processes. The *failure detector* also has a function for managing the process monitor.

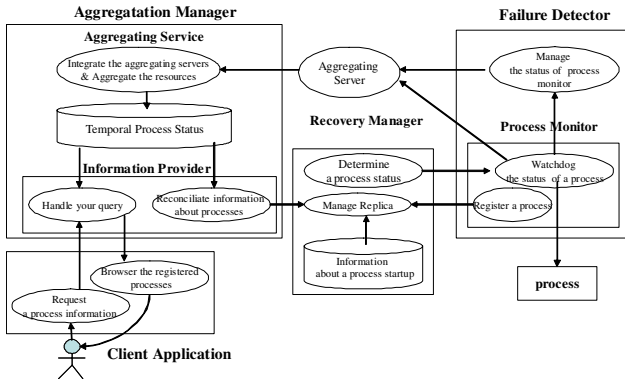


Fig. 3. The system structure of *Hydra*

Command	User	PID	Status	RunTime	Mem
java:Process1:1238	jk	4463	Terminated	0:00	2064K/13114/5
java:Process1:1232	jk	21942	Terminated	0:00	2064K/13114/42
java:Process1:1232	jk	24672	Terminated	0:00	2064K/13114/39
java:Process1:1238	jk	24309	Terminated	0:00	2064K/13116/37

Fig. 4. The processes status when the process (PID=24309) has been killed

Command	User	PID	Status	RunTime	Mem
java:Process1:1238	jk	4463	Terminated	0:00	2064K/13116/5
java:Process1:1232	jk	21942	Terminated	0:00	2064K/13114/42
java:Process1:1232	jk	24672	Terminated	0:00	2064K/13114/39
java:Process1:1238	jk	26736	Running	0:00	2064K/13116/37

Fig. 5. The processes status when the process (PID=26736) has started

The *recovery manager* restarts a process which was failure. Restarting a process, the *recovery manager* co-operates with the *process monitor* which locates the run script for the re-booting procedure of the process. The *process monitor* is clearly separate from a process so that an application in a distributed system cannot consider the failure handling in itself.

Figure 4 shows the information about the status of processes which are registered into the *recovery manager* of *Hydra*. The *failure detector* sends the *aggregating server* a process status. The process status is gathered by the *aggregating server*.

Figure 5 shows the processes status changed into a running status when a process is restarted by the *recovery manager*. To restart a process which was failure, the *recovery manager* refers information about its a startup script with a command path. We can inspect the PID of a restarting process, which was renewed as 26736 from 24309.

5 Conclusion

In this paper we focused on a management of controlling and monitoring of processes in a distributed system based on Linux and UNIX operating system. We described the design and implementation of failure detection for widely distributed systems through Web Services. The failure detector is leveraging Web Service with high interoperability. By basing the service on unreliable failure detector it is clearly separate between failure detection and applications in a distributed system. In our work to date, we have demonstrated that this service can be used by applications no modification to application structure.

References

1. W3C, *Simple Object Access Protocol (SOAP) 1.1*, 2000. URL: <http://www.w3c.org/TR/SOAP>.
2. E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, *Web Services Description Language (WSDL) 1.1*. W3C, 1.1 ed., March 2001. URL: <http://www.w3c.org/TR/wsdl>.
3. S. Chatterjee and J. Webber, *Developing Enterprise Web Services: An Architect's Guide*. Prentice Hall PTR, 2004.
4. N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc., 1996.
5. T. Chandra and S. Toueg, "Unreliable failure detector for asynchronous distributed systems," in *Proceedings of the 10 th Annual ACM Symposium on Principles of Distributed Computing*, August 1991.

CoopStreaming: A Novel Peer-to-Peer System for Fast Live Media Streaming

Jianwei Yin, Weipeng Yao, Lingxiao Ma, and Jinxiang Dong

Department of Computer Science,
Zhe Jiang University, Hang Zhou, China
yao_weipeng@yahoo.com, lma@arcsoft.com.cn,
{zjuyjw, djx}@cs.zju.edu.cn

Abstract. The high bandwidth required by live media streaming limits the number of users, and the quality-of-service (QoS) for media streaming is usually unsatisfactory. This paper presents a novel P2PStreaming model: CoopStreaming (Cooperative Streaming). CoopStreaming tries to achieve a better trade-off, by leveraging the advantages in both P2P and C/S. The core operations in CoopStreaming are very simple: every node periodically exchanges data availability information with a set of partners, and supplies available data to partners. We emphasize four salient features of CoopStreaming: 1) easy to implement, as it does not have to construct and maintain a complex application multicast tree; 2) robust and resilient, as the partnerships enable adaptive and quick switching among multi-suppliers; 3) scalability, as media data distributed in a P2P manner, more users, more quickly. 4) heterogeneity adaptive, as deployed the Progressive Fine Granularity Scalable (PFGS) coding. Extensive simulation results show that the average latency, control overhead, and video quality in our model are all better than those in gossip-based network and tree-based network.

1 Introduction

Multimedia streaming over the Internet is booming nowadays. However, the steaming quality is generally unsatisfactory. For large-scalable broadcasting, the traditional Internet model of point-to-point unicast communication is not enough. So the networking research community proposed IP multicast, a network layer service that allows a single source to distribute a data stream to many simultaneous receivers in an efficient manner. However, this network layer approach has met with limited success due to a number of factors including complexity of network and scalable transport protocols for reliability and congestion control.

In this work, we design CoopStreaming (Cooperative Streaming), which is a novel P2P live media streaming system. Compared with the existed systems [2], CoopStreaming has many new features. First, it adopts many technologies that BitTorrent uses, like send-driven, tif-for-tat, choking and so on. Second, it uses dynamic bandwidth control which adopts a measurement-based optimization techniques for bandwidth-demanding[3]. Three, CoopStreaming uses C/S method to organize nodes, and uses P2P method to distribute data. The simulation results show that we find a very good combination point between C/S mode and P2P mode.

2 Related Work

In this section, we give a brief overview of the existing P2P streaming systems.

A. Fixed-topology Protocols and Systems

These protocols and systems are based on building stable and structural network topology for distribution media streaming. This method usually constructs and maintains a very distribution tree or forest among the nodes. For example, CoopNet [4], NICE [5], ZIGZAG [6], SplitStream [7] and SpreadIt [8] all belong to this category. However, the tree or forest structure is either not very appropriate for enduring frequent breaks in the highly dynamic Internet.

B. Non-topology Protocols and Systems

Such protocols and systems can be classified into two categories: look-up selecting and random selecting. In the former, it uses different look-up algorithms to select the routing of streaming, and the bottleneck lies on the implementing efficient of them. Such protocols include CAN [9], Chord [10], and Pastry [11]. They locate the peers in limited steps. This kind of system includes CFS [12] and PAST [13]. In the latter, the random selecting algorithm can give a better robust and adaptive. CoolStreaming [14] is a representative system. It uses gossip-based protocol [15], and can achieve a similar BitTorrent effect. It uses enough starting delay to guarantee reliability of video QoS. Then CoopStreaming solves the problem.

3 Model

In CoopStreaming, P2P mode is the complement of traditional C/S mode, not to instead of it. Every peer maintains a *PartnerList*. Specially, the server is the partner node of all the peers. Every peer exchanges its Buffer Map (BM) information with its partners if the BM changes. In figure 1, broken lines present the transferring of the control information, real lines present the transferring of the media data. Peer2, Peer3, Peer4 and the server are all the partner nodes of Peer1. Peer1 obtains the data from these non-fixed partners. The node schedule policies are as follows:

- (1) *Node Join*: Once receiving the join request of a new node, the server will put the new node into the *PeerList*, and randomly select a fixed number of nodes as the initial partners of the new node.
- (2) *Node Normally Leave*: When receiving the leave request of a node, the server will simply delete the node from the *PeerList*.
- (3) *Node Unnormally Leave*: Every node must periodically send a message to the server to note that the node is normally working. If the server doesn't receive the message, it will consider the node has unnormally left, and delete it.

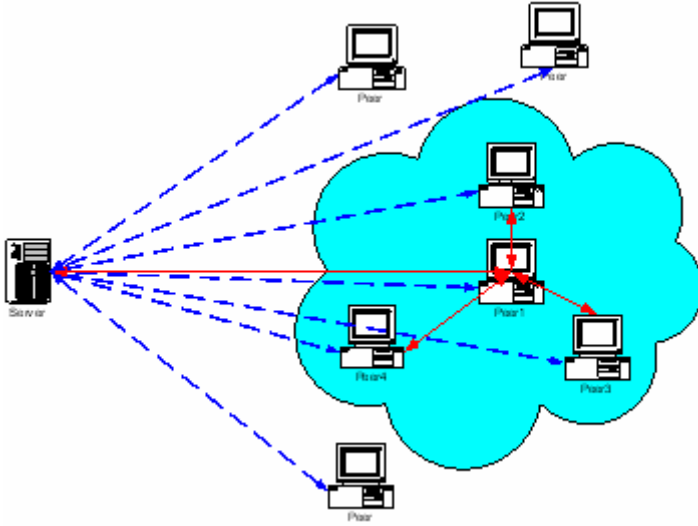


Fig. 1. CoopStreaming Model

3.1.1 Architecture

The client includes two modules: *Data Receive* and *Node Service*. Node Service Module manages distributing media data. Data Receive Module manages receiving media data from other peers. Its functions are as follows:

- (1) *Partner Manage and BM Update*: Dynamically maintaining *PartnerList* and updating partner BM.
- (2) *Data Distribute Algorithm*: Looking up an optimized provider for every data segment according as partner BM.
- (3) *Data Receive*: Receiving data from multi peers by the scheming result.

Also there are a network monitoring module which feedbacks the real-time bandwidth and a QoS controlling module which dynamically adjusts the QoS of users.

3.1.2 Partner Manage and Optimize

After joining in the network overlay, a node first gets an initial *PartnerList* from the server and dynamically updates it. A node will complement new nodes from the server as its new partners. There are two reasons:

- 1) The nodes in the network overlay are very unstable. Once detecting that some nodes have left, the node must get some new nodes from the server.
- 2) A good partner should have a higher uploading bandwidth and the physical link between the node and the partner should be as short as possible. In every round, a node should delete the partner that has the lowest point, and get a new node from the server as its new partner.

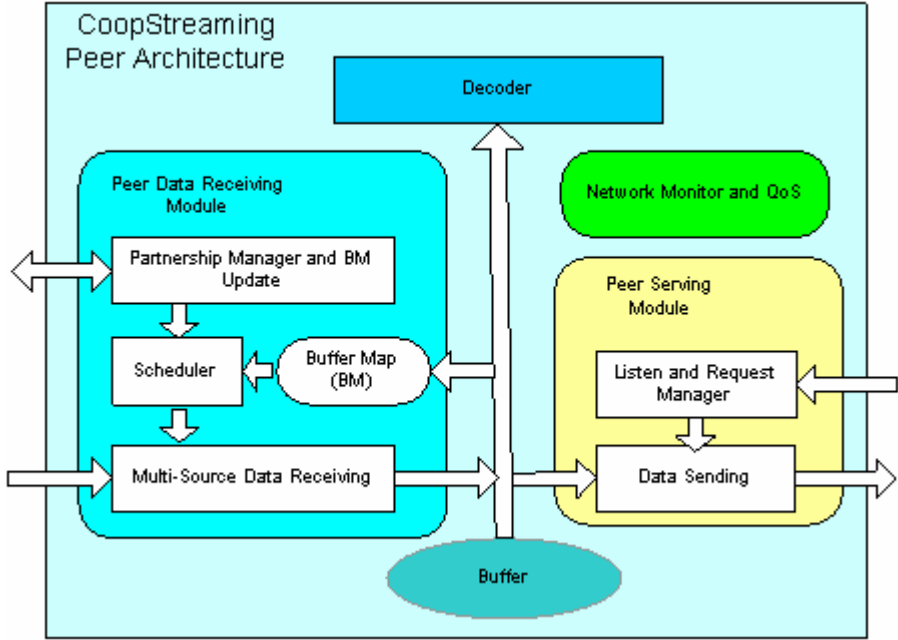


Fig. 2. Client Architecture

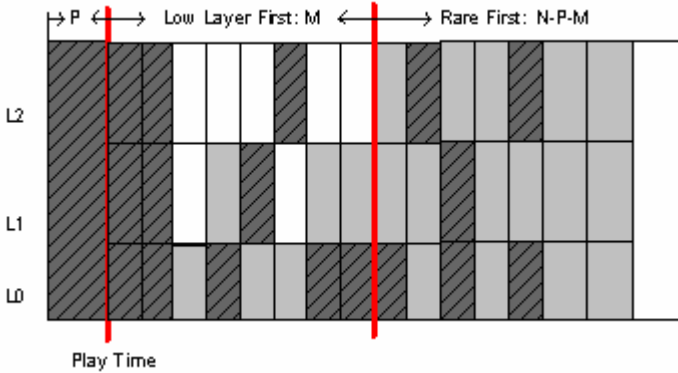


Fig. 3. A Sliding Window For A Moment

3.1.3 Data Scheduling Strategy

The purpose of scheduling is to solve that according as the BM information of the node and its partners, how to fast get media from partners. The scheduling algorithm adopt some principles as follows:

- 1) *Low Layer First*: The low layer data must be transferred first. It's because the dependency of the low layer data to the high layer data in PFGS.

- 2) *Rare First*: In the network overlay, a segment's providers are more less, it will satisfy the deadline more difficultly. So peers will first get the data segment that has only one provider, then get the one that has two providers, and do on the analogy of this. If some segments have the same number of providers, they should select the one has higher upload bandwidth.
- 3) *Random First segment*: An exception to rare first is when the new node just joins in the system. At that time, it hasn't any data to provide to other nodes, so it's important to get a segment of data as soon as possible. For this reason, peers randomly select their first data segment to download until the first segment finishing download, then the strategy changes to the other two principles.

3.1.4 QoS Control

In CoopStreaming, the QoS control adopts the method based on the layer-detect. If current layers don't detect congestions, client will receive a higher layer data. If detecting, client will quit the current top layer. Congestion detect uses the Ratio of Miss (RM) as the criterion, viz. the ratio of non-receiving data in a period of time.

4 Performance Evaluation

We use Matlab for simulation, and we use GT-ITM [16] to generate a transit-stub network topology with about 1000 nodes. It is a cubic Euclidean Space. Nodes are randomly located in the cubic. In this model, we can control the topology maps with different parameters, and randomly select the partners in a 2-D Euclidean space.

We compare our system with tree-based model and gossip-based model. A tree-based model has several hierarchies. A gossip-based model uses a predigested group protocol to implement. The results show that the average latency, control overhead and video quality of our model are a little better than the upper two models.

5 Conclusion and Future Work

This paper presents the design of CoopStreaming. It has the advantage of both administrative management and peer-to-peer system, and composes a efficient system for fast live media streaming. Our extension experiment proves that our design and implement can get a better effect than some current existed protocols and algorithm.

In the future work, CoopStreaming will refer to video-on-demand (VoD) and support VCR-like operations, such as Pause, Resume and Seek. Now CoopStreaming is only a bi-synchronization system. Its sliding windows are the same. But in the VoD system, every user starts at different time. So it may need larger buffers in hard disk. Also we will study the P2P Streaming technology in wireless environment.

References

1. Bram Cohen: Incentives Build Robustness in BitTorrent. First Workshop on Economics of Peer-to-Peer Systems 2003, May 2003
2. Suman Banerjee, Bobby Bhattacharjee: A Comparative Study of Application Layer Multicast Protocols. Submitted for Publication, October 2002

3. T. S. Eugene Ng, Yang-hua Chu, Sanjay G. Rao, Kunwadee Sripanidkulchai, Hui Zhang: Measurement-Based Optimization Techniques for Bandwidth-Demanding Peer-to-Peer Systems. In Proc. of IEEE INFOCOM'03, San Francisco, CA, USA, April 2003
4. V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai: Distributing streaming media content using cooperative networking. In Proc. of NOSSDAV'02, USA, May 2002
5. S. Banerjee, B. Bhattacharjee, C. Kommareddy, and G. Varghese: Scalable application layer multicast. In Proc. of ACM SIGCOMM'02, pages 205–220, Pittsburgh, PA, USA, August 2002
6. D. Tran, K. Hua, and T. Do: Zigzag: An efficient peer-to-peer scheme for media streaming. In Proc. of IEEE INFOCOM'03, San Francisco, CA, USA, April 2003
7. M. Castro, A. Druschel, P. Kermarrec, A. Nandi, A. Rowstron, and A. Singh: SplitStream: High-bandwidth content distribution in a cooperative environment. In Proc. of 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03), Berkeley, CA, USA, February 2003
8. H. Deshpande, M. Bawa, and H. Garcia-Molina: Streaming live media over peer-to-peer network. Technical report, Stanford University, 2001
9. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker: A scalable content-addressable network. In Proc. of ACM SIGCOMM'01, San Diego, CA, USA, August 2001
10. I. Stoica, R. Morris, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In Proc. of ACM SIGCOMM'01, San Diego, CA, USA, August 2001
11. A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proc. of 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany, November 2001
12. F. Dabek, M. Kaashoek, D. Karger, D. Morris, and I. Stoica: Wide-area cooperative storage with CFS. In Proc. of ACM SOSP, October 2001
13. A. Rowstron and P. Druschel. Storage management in past, a large-scale, persistent peer-to-peer storage utility. In Proc. of 18th ACM Symposium on Operating Systems Principles (SOSP'01), Chateau Lake Louise, Banff, Canada, October 2001
14. X. Zhang, J. Liu, B. Li, and T.-SP Yum: CoolStreaming/DONet: A Data-driven Overlay Network for Live Media Streaming. IEEE INFOCOM'05, Miami, FL, USA, March 2005
15. A. J. Ganesh, A.-M. Kermarrec, and L. Massoulié: Peer-to-peer membership management for gossip-based protocols. IEEE Transactions on Computers, 52(2), Feb. 2003
16. Ellen W. Zegura, Ken Calvert, and S. Bhattacharjee: How to model an internetwork. IEEE INFOCOM, San Francisco, CA, 1996

Ontology-Based HTML to XML Conversion

Shijun Li^{1,2}, Weijie Ou¹, and Junqing Yu³

¹ School of Computer, Wuhan University, Wuhan 430072, China
shjli@public.wh.hb.cn, ouwj1981@sohu.com

² Laboratory of Computer Science, Institute of Software,
Chinese Academy of Science, Beijing 100080, China

³ College of Computer Science & Technology,
Huazhong University of Science & Technology, Wuhan 430074, China
yjqinghust@126.com

Abstract. Current wrapper approaches break down in extracting data from differently structured and frequently changing Web pages. To tackle this challenge, this paper defines domain-specific ontology, captures the semantic hierarchy in Web pages automatically by exploiting both structural information and common formatting information, and recognizes and extracts data by using ontology-based semantic matching without relying on page-specific formatting. It is adaptive to differently structured and frequently changing Web pages for a domain of interest.

1 Introduction

Efficiently extracting data from Web pages and converting them into XML have many potential applications [6]. Unfortunately, since most Web pages are in HTML, which is not designed to describe the content of data, there is no simple way to capture the semantic content in Web pages. Current wrapper approaches work well in extracting data from Web sources with the same structure, or with large sets of pages generated using a common template by searchable databases online [1,10], but break down in extracting data from Web sources with differently structured and frequently changing pages for a domain of interest.

In this paper, we study the problem of extracting data from the differently structured and frequently changing pages for a domain of interest. As XML describes the contents of the data and can be queried and processed by a database application [9], we use XML to represent the captured data. We define the domain-specific ontology as the shared conceptualization of knowledge in a particular domain, which consists of thesaurus, label, keywords, begin-landmark, end-landmark, match and separator. According to HTML grammar, an HTML document consists of a sequence of *blocks* that may be *heading*, *paragraph*, *list*, and *table* etc., which may not be in the same semantic hierarchy. As a *heading* element briefly describes the topic and semantic hierarchy of the section it introduces, we can capture the inter-block semantic hierarchy of an HTML document according to different levels of *heading* in it. In such cases of using common formatting information, rather than using *heading* to show the semantic hierarchy,

we can capture the inter-block semantic hierarchy by the common formatting information. After capturing the inter-block semantic hierarchy, we exploit the structural information that HTML tags imply, such as a *list* consists of *items*, to capture the intra-block semantic hierarchy. After the semantic hierarchy structures in Web pages be captured, then we can recognize and extract concepts by using ontology-based semantic matching.

The rest of the paper is organized as follows. Section 2 defines domain-specific ontology. Section 3 and Section 4 present the approach to capturing inter-block and intra-block semantic hierarchy in Web pages, respectively. Section 5 discusses related work. Finally we conclude in Section 6.

2 Domain-Specific Ontology

Domain-specific ontology is the shared conceptualization of knowledge in a particular domain. It consists of concepts and relations among the concepts. We formally define domain ontology as follows.

Definition 1. *Domain ontology O_D is denoted by $O_D(C, R)$, where C is the set of concepts in domain D , R the set of relation among the concepts in C , $R = \{(c_1, c_2, r) \mid c_1 \in C, c_2 \in C, \text{ where } r \text{ is the relation name of } c_1 \text{ and } c_2\}$.*

A Web page is a string of tokens. We define a token to be a word, an HTML token or a punctuation, and a string to be a string of tokens. As different Web page may use different words to express the same concept in a domain, we need thesaurus of a concept and global name for each concept in a domain. And in order to recognize a concept in a Web page, we use *thesaurus*, *label*, *keywords*, *begin-landmark*, *end-landmark*, *match* and *separator*, which are page-specific independent and are formally defined below, to relate a concept.

Definition 2. *A concept c in a domain is denoted by: $c = (\text{name, thesaurus, label, keywords, begin-landmark, end-landmark, match, separator})$, where *name* is the global name of the concept; *thesaurus* is the set of all the synonym corresponding to the concept; *label* and *keyword* are set of strings to mark a nested concept name and the content of a concept, respectively; *begin-landmark* and *end-landmark* are the set of strings that mark the begin and end of the concept, respectively, where we use b, p, t to denote blank, punctuation, and HTML tags; *match* is a regular expression or a file name used to match the concept; *separator* is set of strings to separate the instances, if the concept has multiple instances. All the attributes except *name* are optional.*

In this paper, we use XML DTD to represent domain ontologies. A concept c is called a primary key concept, if it can distinguish different instances of the root concept in a domain ontology. For example, in the domain ontology of home page, *name* is the primary key concept (suppose there is no duplicate name). And for the concept of *ResearchInterest*, *thesaurus* = {research interest; domain of research; recent research}, *label* = {research interest; domain of research; recent research}, *end-landmark* = {b, t}, *separator* = {"", ";"}.

is called a simple string if it contains no labels and keywords of any concept in a domain ontology that marks nested concepts. To capture the nested concepts, we introduce three operators related to a domain ontology as follows.

Definition 3. Let s be a string, the domain ontology O_D . Then, $\gamma_D(s) = c$, if $\exists c \in C$ and $\exists s'$, where s' is a sub-string of s , and $s' \in c.thesaurus$ ($c.thesaurus$ denotes the thesaurus attribute of c); otherwise, $\gamma_D(s) = s$. If there is a substring s' in s that match the $c.match$ of a concept c , then $\mu_c(s) = s'$. Let l be a label or key of a concept c in the domain ontology. Then $\alpha_D(l) = c$.

For example, let $s =$ recent publications, and *publishing* is a concept name in the domain of home pages, then $\gamma_D(s) =$ publishing; let $l =$ recent areas of research, then $\alpha_D(l) =$ ResearchInterest; let $s =$ Tom Bush's home page, then $\mu_{name}(s) =$ Tom Bush. In our system, a domain ontology may be provided manually by system users, domain experts, knowledge engineers, or automatically generated based on induction. It is generated once, but can be used many times.

3 Capturing Inter-block Semantic Hierarchy

3.1 Exploiting Structural Information

An HTML document consists of a *head* element and a *body* element, and must have a *title* element in the head section. We capture the matched content of title as the content of the key concept in the domain ontology as follows.

Let D be an HTML document as follows: $D = \langle html \rangle \langle head \rangle \langle title \rangle T \langle /title \rangle \dots \langle body \rangle B \langle /body \rangle \langle /html \rangle$. Then $\psi(D) = \langle R \rangle \langle K \rangle \mu_K(T) \langle /K \rangle \psi(B) \langle /R \rangle$, where R, K are the name of the root element and the name of the key concept in the domain ontology, respectively, and ψ the converting function that converts an HTML document into an XML one.

The *body* element consists of a sequence of *blocks* that may be *heading*, *paragraph*, *table*, and *list* etc., but the sequence of blocks that are partitioned according to HTML grammar may not be in the same semantic hierarchy. As most HTML documents use *heading* blocks to express its semantic hierarchy, whose six levels from the most important (*h1*) to the least important (*h6*) briefly describe the topic and semantic hierarchy of the block it introduces, we can use it to capture the semantic hierarchy of the sequence of blocks by converting the content of each of the most important *heading* into an XML element, and convert the block that the *heading* introduces as its content as follows, recursively.

Let D be an HTML section: $D = S_0 \langle hn \rangle T_1 \langle /hn \rangle S_1 \dots \langle hn \rangle T_m \langle /hn \rangle S_m$, where hn is the most important heading in D , and each of S_0, \dots, S_m is a sequence of HTML blocks. Then $\psi(D) = \psi(S_0) \oplus \psi(S'_1) \oplus \dots \psi(S'_i) \oplus \dots \psi(S'_m)$, where \oplus denotes concatenate operation of two strings, $S'_i = \langle hn \rangle T_i \langle /hn \rangle S_i$, and $i = 1, \dots, m$. If T_i is a simple string without link then $\psi(S'_i) = \langle \gamma(T_i) \rangle \psi(S_i) \langle / \gamma(T_i) \rangle$; If T_i is a simple string but with a link point to a page, suppose the body of this page is B , then $\psi(S'_i) = \langle \gamma(T_i) \rangle \psi(\langle B \rangle) \oplus \psi(S_i) \langle / \gamma(T_i) \rangle$; If T_i is not a simple string, that is, it contains nested concepts recognized by labels or keywords, then $\psi(S'_i) = \psi(T_i) \oplus \psi(S_i)$.

The above converting function is recursive, we can recursively apply it till the whole semantic hierarchy in a Web page is captured.

3.2 Exploiting Formatting Information

Some Web pages may use formatting information, rather than using any *heading* blocks to show their semantic hierarchy. The formatting information that can be exploited for capturing semantic hierarchy includes *bold* tags, *strong* tags and the biggest font etc. In such case, the key is to find out all the text strings that have the most important formatting information, whose function is just as the most important *heading* in an HTML document. Take it as the important heading elements, then we can use the above approach to capture the semantic hierarchy and convert it into XML. Note that we can recursively use the strongest formatting information and take it as the heading elements, so that we can capture the whole semantic hierarchy in a Web page.

4 Capturing Intra-block Semantic Hierarchy

4.1 Paragraphs

After recursively capturing the inter-block semantic hierarchy, the next step is capturing the intra-block semantic hierarchy. First, consider the p element. If the content of a p element is a simple string, we need do nothing. Otherwise, the content of a p element contains nested concepts marked by labels and keywords in domain ontology. We capture the nested concept of p element as follows.

Let $D = \langle p \rangle S \langle /p \rangle$. If S is a simple string, then $\psi(D) = s$. Otherwise, if S contains nested concepts, let $S = S_1CS_2$, if C is a keyword related to a concept c , then $\psi(D) = \psi(S_1) \oplus \langle c \rangle C \langle /c \rangle \oplus \psi(S_2)$; if $C = C_1TC_2$, where C_1 and C_2 are the label and the separate related a concept c in domain ontology, respectively, then $\psi(D) = \psi(S_1) \oplus \langle c \rangle \psi(T) \langle /c \rangle \oplus \psi(S_2)$.

4.2 Lists

In HTML documents, both unordered lists *ul* and ordered lists *ol* consist of *li* elements, which are their items. All the *li* elements logically belong to the same level, and can be taken as the children elements of its parent concept. So we can convert all the *li* elements into the children concept of the parent concept in the domain ontology as follows.

*Let $L = \langle ul \rangle L_1 \dots L_n \langle /ul \rangle$, or $L = \langle ol \rangle L_1 \dots L_n \langle /ol \rangle$, where L_i is a *li* element and $i = 1, \dots, n$. Let the name of the concept recognized before L is b . If b has a children elements c in domain ontology DTD, then $\psi(L) = \psi(L_1) \oplus \dots \oplus \psi(L_n) = \langle c \rangle \psi(s_1) \langle /c \rangle \dots \langle c \rangle \psi(s_n) \langle /c \rangle$; Otherwise, $\psi(L) = \psi(L_1) \oplus \dots \oplus \psi(L_n) = \langle b \rangle \psi(s_1) \langle /b \rangle \dots \langle b \rangle \psi(s_n) \langle /b \rangle$, where s_1, \dots, s_n is the content of L_1, \dots, L_n , respectively.*

4.3 Tables

In Web pages, table cells generally contain heading information via the *th* element and data via the *td* element, and may span multiple rows and columns. If a *th* or *td* element contains *colspan* = n or *rowspan* = n , the particular cell of the *th* or *td* is to be expanded to $n - 1$ more columns, starting from the current cell in the current row, or to the next $n - 1$ rows in the current column, respectively. By inserting redundant cells according to the attributes of *colspan* and *rowspan*, we can normalize an HTML table, in which each row has the same number of cells aligned. The attribute-value pairs in the normalized table is what we want to captured. The mapping rule is suited for the HTML tables with marked headings via *th* elements. To convert HTML tables without marked headings via *th* elements, we can recognize their headings by formatting information, which are mainly font (size, bold) [4]. In the case of tables are for creating some type of multiple-column layout for easy viewing, we just ignore the table tags.

4.4 Frames

HTML frames allow authors to present documents in multiple views. An HTML document that describes frame layout (called a frameset document) has a HEAD, and a FRAMESET in place of the BODY. To capture the semantic hierarchy in a frame Web pages equals to capture each frame in the page.

*Let $D = \langle \text{frameset} \dots \rangle \langle \text{frame} = \text{"frame1.html"} \rangle \dots \langle \text{frame} = \text{"framen.html"} \rangle \langle \text{/frameset} \rangle$, and f_1, \dots, f_n denote *frame1.html*, ..., *framen.html*, respectively. Then $\psi(D) = \psi(f_1) \oplus \dots \oplus \psi(f_n)$.*

5 Related Work

The manually or semiautomatic approaches [8], and the induction approaches [3] to extract wanted data from a Web site work well in extracting data from multiple Web pages with the same structure, but break down when web pages changed and for large scale of differently structured web pages. The ontology-based or concept schema-based approaches [2,7,5] create a domain-specific wrapper called an extraction ontology, rather than creating a page-specific wrapper. These approaches work well in the regularly formatted pages such of news papers Web sites, but breaks down in the domain where the page formatting can be more complex such as the domain of home pages. These approaches also partly rely on the page-specific formatting. In this paper, we improve our earlier work [4] by using ontology. A system based on the approach presented here has been implemented. We have performed experiment to extract data on the home pages of researchers in universities with which existed related approaches have poor performance. The domain ontology of home pages is generated easily by manually. The average precision (number of correctly extracted concepts / number of all extracted concepts) and the recall (number of correctly extracted concepts / number of all the concepts that should be correctly extracted) of our approach are approximately 81%, and 74%, respectively, which are calculated by manually

based on experiment data on randomly chose 100 home pages. Compared with other related approaches, our approach has the following advantages.

(1) It does not rely on any page-specific formatting to guide extraction, and need no any training on the format of those pages. It is adaptive for loosely structured pages and resilient to changes in pages for a domain of interest.

(2) It exploits both formatting and structural information implied by HTML tags to automatically capture the semantic hierarchy, and finer granularity information in Web pages based on domain ontology. Moreover, it uses recursive function, so it can handle arbitrarily nested structure in Web pages.

6 Conclusion

This paper defines the domain-specific ontology, exploits both structural information and common formatting information to automatically capture the nested semantic hierarchy, and recognize and extract data in Web pages by using ontology-based semantic matching. It is adaptive with loosely structured pages and resilient to changes in pages for a domain of interest.

References

1. Arvind Arasu and Hector Garcia-Molina. Extracting Structured Data from Web Pages. In Proc. VLDB (2003) 337-348
2. David W. Embley, Cui Tao, and Stephen W. Liddle: Automatically Extracting Ontologically Specified Data from HTML Table of Unknown Structure. In ER 2002 (2002) 322-337
3. Nicholas Kushmerick, D. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In IJCAI (1997), 729-737
4. Shijun Li, Mengchi Liu, Tok Wang Ling, and Zhiyong Peng. Automatic HTML to XML Conversion. In Proc. WAIM (2004), 714-719
5. Xiaofeng Meng, Hongjun Lu, Haiyan Wang, and Mingzhe Gu. Data Extraction from the Web Based on Pre-Defined Schema. *J. Comput. Sci. Technol.*, 17(4)(2002):377-388
6. Zhiyong Peng, Qing Li, Ling Feng, Xuhui Li, and Junqiang Liu. Using Object Deputy Model to Prepare Data for Data Warehousing. *IEEE Transaction on Knowledge and Data Engineering*, Vol. 17, NO. 9, (2005)
7. Thomas E. Potok, Mark T. Elmore, Joel W. Reed, and Nagiza F. Samatova. An Ontology- Based HTML to XML Conversion Using Intelligent Agents. In J.Loyd et al., editor, HICSS (2002), 120-129.
8. Arnaud Sahuguet and Fabien Azavant. Building Intelligent Web Applications Using Lightweight Wrappers. *Data and Knowledge Engineering* (2001), 36(3): 283-316
9. Guoren Wang, Bing Sun, Jian-Hua Lv, and Ge Yu. RPE Query Processing and Optimization Techniques for XML Databases. *J. Comput. Sci. Technol.*, 19(2)(2004):224-237
10. Wensheng Wu, Clement T. Yu, AnHai Doan, and Weiyi Meng. An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web. In Proc. SIGMOD (2004), 95-106

Image Matrix Fisher Discriminant Analysis (IMFDA)- 2D Matrix Based Face Image Retrieval Algorithm

C.Y. Zhang, H.X. Chen, M.S. Chen, and Z.H. Sun

Inst. of Communication Engineering, Nanling Campus, Jilin Univ., 130025, China
chunyuzhang0320@hotmail.com, chx@jlu.edu.cn,
mianshuchen@sohu.com, sunzh329@yahoo.com.cn

Abstract. Traditional 1D vector based FDA algorithm is popular used in face image retrieval. In FDA, data is represented by 1D vector, which is converted from image matrix. Usually, this conversion makes the number of examples less than that of data dimension, which will give rise to small sample problem. To overcome this problem, 2D matrix based algorithm is proposed, in which the within-class scatter matrix is derived directly from matrix. In the existing matrix based algorithms, IMPCA and GLRAM don't utilize discriminant information between classes. Although TDLDA goes further, yet it is solved by iterative steps. Here we propose a new matrix based technique: IMFDA. It not only takes the advantage of discriminant information between classes, but also can be solved as a generalized eigenvalue problem. Experiments on ORL face database show that the new algorithm is more efficient than IMPCA, GLRAM and TDLDA with lower test error and shorter running time.

1 Introduction

Traditional FDA [1-4] is a classical algorithm for feature extraction and dimension reduction. The main step of FDA is described as follow. First convert image matrix into 1D vector, then construct between-class scatter matrix and within-class scatter matrix, at last find the subspace such that in that space the ratio of the between-class distance to the within-class distance is maximum. Usually the biggest problem with FDA algorithm in face image retrieval is that within-class scatter matrix is singular. The reason for that is that data is represented by vector in traditional FDA, which needs to convert image matrix into 1D vector, and will make the number of samples less than that of data dimension. Moreover, the conversion from image matrix into 1D vector will damage the spatial correlations among image pixels, which present the spatial structure of face image, and also contain much discriminant information for classification.

To overcome the disadvantage of 1D vector based algorithm, a novel data representation model: 2D matrix based model was proposed, in which the data is represented by two-dimensional matrix. Matrix based algorithms had been widely used not only in image classification [5,6,8,9], but also in image compression [7,9], and also recently in representing multidimensional data [10]. The 2D matrix based algorithm was first proposed by Yang J. [5] for the generalization of PCA, named IMPCA (Image Matrix Principle Component Analysis), whose aim was to find a projection matrix X such that in the projected space the variance of total projection is

maximal. Then Ye J.P. [7,9] adopted the same concept and presented GLRAM (Generalized Low Rank Approximations Matrix), which found two projection matrices **L** and **R** by iterative way. From different point of view Shashua A. [8] deduced linear image coding using the concept of Tensor-Rank Principal (TRP), but in principal, optimization problem in GLRAM and TRP is the same. Although these algorithms adopted novel 2D matrix based model, its limitation is still trying to minimize reconstruction error, which is essential for the image compression, but not the real goal of classification, thus it is necessary to use FDA criteria in 2D matrix based model. Although TDLDA (Two-Dimensional Linear Discriminant Analysis) uses discriminant information, but it is solved by iterative steps, suffering from heavy computation load.

In this paper we present a novel approach IMFDA, the novelty lies that it's not only based on the 2D matrix based model, but also let the projection matrix maximize the ratio of the between-class distance to the within-class distance. We conclude with extensive empirical evaluations of IMFDA, showing its advantage over alternative algorithms in lower test error and shorter running time.

The remainder of this paper is organized into four sections. Section 2 describes the proposed algorithm IMFDA. In section 3 we give experimental results. Conclusion can be seen in the last section.

2 IMFDA

This section is composed of two parts, in the first part we introduce the theory of IMFDA algorithm, and in the second part we present the retrieval step.

2.1 IMFDA Feature Extraction Step

IMFDA tends to find the projection matrix such that in the space derived by that matrix the ratio of the between-class distance to the within-class distance is maximal. To begin with the description of IMFDA, let's fix some notations. Suppose *C* classes *n* image samples, *i*th class has *n_i* image, **A_{ij}** is the *i*th class *j*th image in the gallery, **Ā** is the mean of all images, **Ā_i** (*i*=1, ..., *C*) is the mean of the *i*th class images. In the space derived by projection matrix **X**, the distance of between-class **TS_b**, the distance of within-class **TS_w** and the sum distance **TS_t** are defined as follow:

$$TS_b = \sum_{i=1}^C n_i (\mathbf{Y}_i - \bar{\mathbf{Y}})(\mathbf{Y}_i - \bar{\mathbf{Y}})^T = \sum_{i=1}^C n_i [(\bar{\mathbf{A}}_i - \bar{\mathbf{A}})\mathbf{X}][(\bar{\mathbf{A}}_i - \bar{\mathbf{A}})\mathbf{X}]^T \tag{1}$$

$$TS_w = \sum_{i=1}^C \sum_{j=1}^{n_i} (\mathbf{Y}_{ij} - \bar{\mathbf{Y}}_i)(\mathbf{Y}_{ij} - \bar{\mathbf{Y}}_i)^T = \sum_{i=1}^C \sum_{j=1}^{n_i} [(\mathbf{A}_{ij} - \bar{\mathbf{A}}_i)\mathbf{X}][(\mathbf{A}_{ij} - \bar{\mathbf{A}}_i)\mathbf{X}]^T \tag{2}$$

$$TS_t = TS_b + TS_w \tag{3}$$

The optimal projection matrix **X** can be found in the following maximization step:

$$J_p(\mathbf{X}) = \arg \max (TS_b / TS_t) \quad \text{or} \quad J_{p1}(\mathbf{X}) = \arg \max ((TS_b / TS_w)) \tag{4}$$

According to Fukunaga K. [1], eigenvectors calculated from above two equations have the same discriminant ability. So in order to simplify, we use $J_\rho(\mathbf{X})$ as main function in our experiment.

As pointed out also in [1], the solutions of the following equations are identical:

$$J_\rho(\mathbf{X}) = \arg \max(\mathbf{TS}_b / \mathbf{TS}_t) = \arg \max(tr(\mathbf{TS}_b) / tr(\mathbf{TS}_t)) \tag{5}$$

And according to the theory of matrix, for an arbitrary matrix \mathbf{B} , exists $tr(\mathbf{BB}^T) = tr(\mathbf{B}^T\mathbf{B})$. For simplification we define the following matrixes: Image Total Scatter matrix \mathbf{G}_t and Image Between-class Scatter matrix \mathbf{G}_b :

$$\mathbf{G}_b = \sum_{i=1}^C n_i [(\bar{\mathbf{A}}_i - \bar{\mathbf{A}})]^T [(\bar{\mathbf{A}}_i - \bar{\mathbf{A}})] \quad \mathbf{G}_t = \sum_{i=1}^C \sum_{j=1}^{n_i} [(\mathbf{A}_{ij} - \bar{\mathbf{A}})]^T [(\mathbf{A}_{ij} - \bar{\mathbf{A}})] \tag{6}$$

So IMFDA is also equal to calculate the following equation:

$$J_k(\mathbf{X}) = \arg \max(\mathbf{X}^T \mathbf{G}_b \mathbf{X} / \mathbf{X}^T \mathbf{G}_t \mathbf{X}) \tag{7}$$

This is a generalized eigenvalue problem. The optimal projection matrix \mathbf{X} can be obtained by calculating the first k largest eigenvalues and corresponding eigenvectors of $(\mathbf{G}_t)^{-1} \mathbf{G}_b$. In general, \mathbf{G}_t is a non-singular matrix and has lower dimension compared with the conventional total scatter covariance matrix.

2.2 IMFDA Retrieval Step

Each image \mathbf{A}_{ij} is mapped into projection matrix \mathbf{X} , and coefficient matrix \mathbf{Y}_{ij} is obtained. We can directly use NN classifier in the projected space, by comparing coefficient matrix \mathbf{Y}_{ij} . For a given test image \mathbf{A}_t , its' coefficient matrix is \mathbf{Y}_t , then in NN classifier:

$$dis = \min(\|\mathbf{Y}_t - \mathbf{Y}_{ij}\|_2) \tag{8}$$

That is to say we only need to calculate the distance between different coefficient \mathbf{Y}_t and \mathbf{Y}_{ij} , the best class prediction of \mathbf{A}_t is the class which the minimum dis value \mathbf{A}_{ij} belongs to.

3 Experimental Results

In this section we conducted experiments to test the performance of our algorithm by comparing it with other algorithms, including matrix based algorithms: IMPCA [5] GLRAM [7] and TDLDA [9], and vector based algorithms: PCA [2] and FDA [3]. A NN classifier is employed to classify in the projected feature space.

The experiments were based on a widely used pattern recognition benchmark database: ORL face dataset. This dataset includes forty distinct subjects and each subject has ten images with resolution of 112×92 . All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position. We conduct experiment ten times and each time randomly select five images of each

person for training and the remaining five for testing (training set and testing set both has 200 images), there is no overlap between the two sets. We treat each test as a Bernoulli random test and give the average test error number, which is defined as the number of wrong classification.

4.1 Test Error

First we give the test error number of matrix based algorithms in Table 1. Because the test error number depends highly on the number of discriminant vector k in projection matrix X , we let k vary from 1 to 15. For vector based algorithms according to VC dimension theory the optimal k is about $C-1$ (equals to 39 in ORL dataset).

Table 1. Error number of IMPCA, GRLAM and IMFDA Table 1

Algorithm	Number of the projected vector k														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
IMPCA	26	14	14	13	13	13	13	12	13	13	13	13	13	13	13
GLRAM	176	79	28	19	15	13	14	14	13	13	13	13	13	13	12
TDLDA	186	108	64	38	28	19	16	14	14	15	13	14	15	15	15
IMFDA	17	12	11	11	11	11	10	10	10	10	10	10	11	11	11

From table 1, we can say:

- Even when k is small the test error number is low, and when k is above 6, the error numbers of all algorithms are almost stable, this is mainly because the first 6 largest eigenvalues already take the majority energy of G , [6].
- According to test error number IMFDA is the best among 2D matrix based algorithms. One reason is that IMFDA uses discriminant information between different classes, whereas IMPCA and GLRAM are just trying to minimize the reconstruction error. Although 2DLDA algorithm takes advantage of the classification information as IMFDA used, it is difficult for the user to set parameters in TDLDA.

The optimal performance of vector based algorithms PCA and FDA is given in Table2.

Table 2. Optimal test error number of PCA and FDA

	PCA	FDA
Optimal test error number	29	20

Compare Table 1 with Table 2, it is obviously to say, matrix based algorithms can always get better result. This is because the 2D matrix model retains the spatial correlations between image pixels, which contains the structure information of face image, are still useful for classification problem. Another reason for the higher performance is, in 2D matrix model, the size of Image Total Scatter matrix G , and Image Between-class Scatter matrix G_b , are small, we can calculate corresponding

eigenvectors more accurately. Furthermore the defined matrix G_r and G_b is mainly based on the column of original matrix, so this kind of algorithms are more powerful to tolerate the variation in face images, consequently are more likely to get more stable recognition ratio in different face datasets.

4.2 Computation Time

Feature extraction and classification time of different algorithms are given in Table 3.

Table 3. Running time of different algorithms

Algorithms	Feature extraction time (s)	Classification time(s)
PCA	382.6	4.8
FDA	391.4	4.9
IMPCA	37.1	2.6
GLRAM	41.9	0.6
TDLDA	56.7	0.6
IMFDA	37.8	2.7

From Table 3, it can be seen that the matrix based algorithms can always have shorter running time compared with vector based algorithms. There are two reasons: Firstly vector based algorithms have a time-consuming step which converts image to vector; Secondly vector based algorithms need more discriminate vectors in projection matrix in order to get lower test error number.

According to section 4.1, we know that IMFDA has the lowest test error number, and almost the second shortest running time, only a little longer than IMPCA.

4.3 Storage

From Table 1, we can see that we need small k to get satisfactory result for 2D matrix based algorithms, consequently we also need less storage space to save them. For example in IMFDA, if we select $k = 6$, the dimension of projection matrix X is 92×6 , but for FDA, dimension is 10304×36 , only the storage of discriminate matrix X , we need 92×6 times more space for FDA than IMFDA. On the other hand, more storage for X , means more storage of projection coefficient and at last longer classification time.

5 Conclusion

In this paper a novel feature extraction 2D matrix based algorithm IMFDA is proposed. Compared with the traditional vector based algorithms, this algorithm has two advantages, first it adopt a new 2D matrix model to represent image data, avoiding the time consuming step of converting the image matrix into vector, furthermore implicitly solve the singularity of within-class matrix which most encountered in small sample problem. Second it uses the information between different classes, through which maximize the ratio of between-class matrix to the

total variance matrix, improves the discriminant performance, so we can conclude that IMFDA is more suitable to image classification compared with other 2D matrix based algorithms, and it will play a very important role in image retrieval system.

References

1. Fukunaga K.: Introduction to Statistical Pattern Classification, Academic Press, San Diego, California, USA, (1990)
2. M.A. Turk and A.P. Pentland. Face recognition using Eigenfaces. International Conference on Computer Vision and Pattern Recognition , (1991) 586–591
3. Belhumeur P. N., Hespanha J. P., Kriegman D. J.: Eigenfaces vs. Fisherfaces: Recognition using Class Specific Linear Projection, IEEE Transaction on Pattern recognition and Machine Intelligence., Vol.19. (1997) 11-720
4. Swets D., weng J.: Using Discriminant Eigenfeatures for image Retrieval, IEEE Transaction on Pattern recognition and Machine Intelligence, Vol.18. (1996).831-836
5. Yang, J. Yang J.-Y.: From Image Vector to Matrix: a straightforward Image Projection Technique—IMPCA vs. PCA, Pattern Recognition, Vol.35. (2002) 1997-1999.
6. Yang J., Zhang D., Frangi A.F., and Yang J.Y.: Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition, IEEE Transaction on pattern analysis and machine intelligence, Vol.26. (2004).
7. PYe J.: Generalized Low Rank Approximations of Matrices, ICML Conference Proceedings, (2004) 887–894
8. Shashua, A., Levin, A.: Linear Image Regression and Classification Using the Tensor-Rank Principle, Conference of Computer Vision and Pattern Recognition, (2001).
9. PYe J., Janardan R., and Li. Q.: Two-Dimensional Linear Discriminant Analysis, Conference of Neural Information Processing System, (2004)
10. Wang, H.C., Ahuja, N.: Compact Representation of Multidimensional Data Using Tensor Rank-One Decomposition, Conference of Computer Vision and Pattern Recognition, (2004).

Approximate Common Structures in XML Schema Matching¹

Shengrui Wang¹, Jianguo Lu², and Ju Wang²

¹ Department of Computer Science, University of Sherbrooke,
Sherbrooke, Québec, J1K 2R1, Canada
shengrui.wang@usherbrooke.ca

² School of Computer Science, University of Windsor,
Windsor, Ontario, N9B 3P4, Canada
jlu@cs.uwindsor.ca, ju_wang@yahoo.com

Abstract. This paper describes a matching algorithm that can find accurate matches and scales to large XML Schemas with hundreds of nodes. We model XML Schemas as labeled, unordered and rooted trees, and turn the schema matching problem into a tree matching problem. We develop a tree matching algorithm based on the concept of Approximate Common Structures. Compared with the tree edit-distance algorithm and other Schema matching systems, our algorithm is faster and more suitable for large XML Schema matching.

1 Introduction

Schema matching is widely studied in database research [1] [5], with the aim to bridge relational and semi-structured data models, or to integrate data with either homogeneous or heterogeneous data models [8]. Schemas are usually modeled as trees and tree matching has inevitably become one of the main issues in Schema matching. A number of intuitively appealing techniques have been proposed including Cupid [5], similarity flooding [6], LSD [2], ISSDE [11], SKAT [7] and COMA [5]. The most formal approach is tree matching based on edit distance. Those for unordered tree matching [10] and [3] are more useful than for ordered tree matching [12]. Unfortunately, matching algorithms for unordered trees suffer from exponential complexity. In practice, researchers propose approximate algorithms. This is the case in [9] and [13] where, under the constraints of ancestor preserving and exact node-to-node mapping, subtree similarities are iteratively calculated in a bottom-up way.

In our work, we also model an XML Schema as an unordered, labeled and rooted tree. We are more interested in the overall similarity between two XML Schemas than the correspondence of individual elements. Our algorithm identifies the structural relations by extracting approximate common substructures in two trees. Several heuristics have been proposed to drastically reduce the search space in order to achieve a good trade-off between matching optimality and time complexity. The modeling phase, the computation of node similarity and some detailed experimental results are described in [4]. The present paper is dedicated to the presentation of the new tree matching algorithm that has been designed for computing the structural similarities.

¹ The work has been supported by NSERC, CITO, NSERC CRD and NCE Auto 21.

2 Approximate Tree Matching

The algorithms discussed in Section 1 are not adequate for two reasons. One is the efficiency problem in dealing with large schemas. Another relates to the constraint imposed by ancestor relation preserving. To understand this latter point, let us consider the two schemas in Figure 1. They are not similar because of the structural difference. However, they do share a common substructure and an approximate common substructure as shown in Figure 2. Their extraction would allow some further interesting comparison between the two Schemas. The goal of our algorithm is to extract a **disjoint** set of the largest approximate common substructures (ACS) between two trees. This set of ACSs represents the most likely matches between substructures in the two schemas. Indeed, the algorithm computes structure similarity for each pair of substructures by comparing “root parts” of subtrees while still taking into account the structure similarity between other parts (i.e. parts closer to leaves). In the follows, $nodeSim(u, p)$ is used to represent the similarity between the nodes u and p .

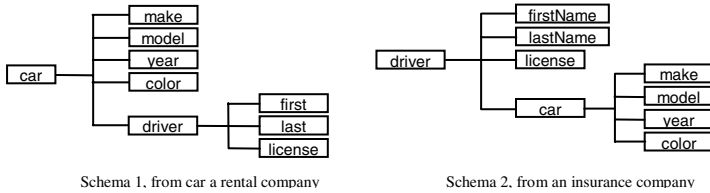


Fig. 1. Car-Driver Schemas

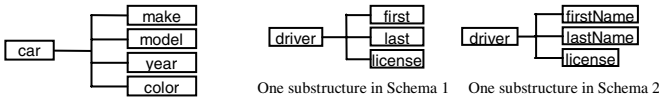


Fig. 2. A common substructure (left) and two similar substructures (middle and right figures)

2.1 Approximate Common Substructure (ACS)

Given two trees T_1 and T_2 , the concept of ACS is related to subtree matching. A quality measure, defined as the *structure similarity*, is necessary to distinguish between a “good” ACS and a “not very good” ACS. Formally, the structure similarity between two substructures, $subStr_1$ from T_1 and $subStr_2$ from T_2 , can be defined as:

$$structSim(subStr_1, subStr_2) = \max_M C(M)$$

where M is any mapping between $subStr_1$ and $subStr_2$ satisfying the following two conditions: 1) $(u, p) \in M, (v, q) \in M$ and $u = v$ imply $p = q$; 2) M is ancestor preserving. The similarity measure $C(M)$ should satisfy the following two conditions: 1) $0 \leq C(M) \leq 1, C(M) < 1$ if M is not an isomorphism between $subStr_1$ and $subStr_2$; 2) $C(M)$ is positively proportional to the size of M and negatively proportional to the number of unmatched nodes. Unfortunately, computing $structSim(subStr_1, subStr_2)$ is NP. In our algorithm, $structSim(subStr_1, subStr_2)$ is replaced by an approximate

similarity function $treeSim()$, while $subStr_i$ are limited to rooted subtrees. Another important measure is the *matching percentage*, $mPer(u, p)$. If we use $ACS(u, p)$ to denote the ACS rooted at u and p , $|ACS(u, p)|$ to denote the number of matched node pairs, then we have $mPer(u, p) = \frac{|ACS(u, p)|}{(|T_1| + |T_2|) / 2} = \frac{2|ACS(u, p)|}{(|T_1| + |T_2|)}$. An optimal ACS

reaches a trade-off between the structure similarity and the matching percentage.

The algorithm aims to find a unified mapping M_{schema} composed of the set of all the mappings derived from **disjoint** ACSs

$$M_{schema} = m_{ACS1} \cup m_{ACS2} \cup \dots \cup m_{ACSi} \cup \dots \cup m_{ACSn}$$

so that each ACS_i has a combined score of the structure similarity and the matching percentage beyond a fixed threshold. It is important to notice that while ancestral relations are preserved within each ACS_i ; they do not have to be preserved in the unified mapping M_{schema} . This is a distinctive feature of the algorithm proposed in this paper that makes the matching of the above car-driver Schemas possible.

2.2 The Matching Algorithm

Given a node pair (u, p) , where $u \in T_1$ and $p \in T_2$ respectively, the structure similarity, $treeSim(u, p)$, between the subtrees rooted at u and p is defined as follows:

$$treeSim(u, p) = \alpha * nodeSim(u, p) + (1 - \alpha) * subTreeSim(u, p)$$

where $subTreeSim(u, p)$ represents the similarity computed based on the subtrees rooted at u and p , and is the major concern of the algorithm. α is a factor whose value is between 0 and 1, reflecting the weight of the two parts. This definition, easily justifiable for a large number of real applications, is deliberately in favor of the root parts of the two subtrees and suggests a recursive approach to matching the two trees. Figure 4 outlines the general idea of the approach adopted for the new algorithm.

The subtree similarity corresponds in fact to the similarity between two forests under u and p . Since the size of each subtree in the forests can be very large, we trim the size of each subtree to two levels by considering each subtree beyond the level 2 as a super-node as shown in the forest under (p) in Figure 3. The procedure for computing $subTreeSim(u, p)$ includes matching trees in the two forests and computing node-to-node similarities. The following conditions have been taken into account: 1) preservation of ancestor relation; 2) if a node u_1 is deleted, all the child nodes of u_1 will be moved up to become children of the parent of u_1 ; 3) if at least one of the two nodes is a super-node, then $forestNodeSim(u_1, p_1) = treeSim(u_1, p_1)$.

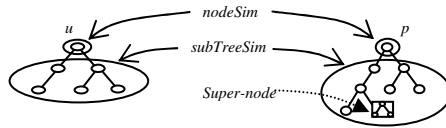


Fig. 3. Basic Idea of the Proposed Matching Algorithm

To compute $treeSim(u, p)$, we distinguish three cases:

Case 1: Both u and p are leaves. In this case, we define

$$treeSim(u, p) = nodeSim(u, p) \dots \dots \dots (1)$$

Case 2: One of the two nodes, u , is a leaf, another one is an inner node, then

$$treeSim(u, p) = \alpha * nodeSim(u, p) + (1 - \alpha) * subTreeSim$$

$$= \alpha * nodeSim(u, p) + (1 - \alpha) * \frac{\beta}{\sqrt{1 + |descendants|}} \dots \dots \dots (2)$$

where $|descendants|$ denotes the number of descendants of the non-leaf node. $\sqrt{1 + |descendants|}$ is a penalty factor that reflects the difference between the two forests under u and p . β is a user-defined parameter, set to 0.3 in our experiments?

Case 3: Both u and p are inner nodes.

$$treeSim(u, p) = \alpha * nodeSim(u, p) + (1 - \alpha) * subTreeSim$$

$$= \alpha * nodeSim(u, p) + (1 - \alpha) * \max_M \left\{ \frac{\sum_{(i,j) \in M} forestNodeSim(i, j)}{\sqrt{1 + |deletedChild|} \cdot NbLF} \right\} \dots \dots \dots (3)$$

In this equation, M is any mapping built following the above conditions 1 and 2, i.e. M is an ancestor order preserving mapping between the remained forest under u and the remained forest under p once certain immediate children of u and p are deleted. $NbLF$ is the number of nodes in the larger (remained) forest. The formula (3) can be interpreted as follows. $(\sum_{(i,j) \in M} forestNodeSim(i, j)) / |M|$ is the average similarity between the matched nodes which will be penalized by two factors: one is related to deleted nodes (division by $\sqrt{1 + |deletedChild|}$) and the other one is related to percentage of non-matched nodes (multiplication by $|M| / NbLF$). This formula materializes the goal of the matching, which is to search the best ancestor order preserving correspondence between the two forests in terms of the similarity and the number, while limiting the number of deletions.

The matching process starts with leaf nodes of the two trees and goes upwards. Each pair of subtrees will be matched after all their pairs of subtrees have been matched. The output of the matching algorithm is all the similarity and the corresponding mapping for each pair of subtrees.

2.3 Identifying ACSs

Both $treeSim(u, p)$ and $mPer(u, p)$ should be considered to determine qualified ACSs for the final mapping M_{schema} . If we project these two values for every node pair into a two-dimension-plane, we will get a scatter chart like the one in Figure 4. Each point denotes the result of an ACS candidate. Point $A(1, 1)$ represents an ideal matching – both the structural similarity and matching percentage reach the maximum value. Obviously, the points near A reflect the good ACS candidates. Points near $D(0, 0)$

represent the poorest candidates – low similarity and low matching percentage, and points near $B(0, 1)$ and $C(1, 0)$ represent those candidates for which only one of the values is high. Generally, most points fall into the area in between.

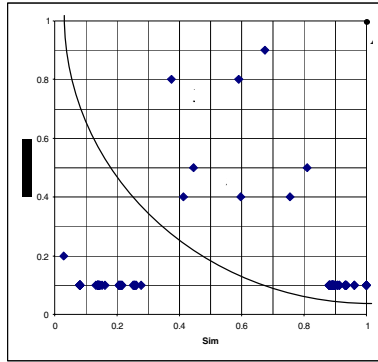


Fig. 4. *treeSim()* and *mPer()* chart (Car-Driver Schemas)

To determine good ACSs from the scatter chart, we use the Euclidian distance from the node to the perfect matching point A . The idea of this approach is shown in Figure 4: the arc represents those points whose distance to point A is equal to the threshold, therefore the points covered by the sector will be considered as admissible candidates to generate the ACSs. Details of the extraction procedure are omitted because of space limit.

2.4 Reporting Results – Mappings and Schema Similarity

Retrieving mappings is relatively straightforward once M_{schema} is identified. Each mapping is reported as two strings in XPath format, i.e. a string of names from the root element (always ‘schema’) to this matched element, and the names are delimited by slash. e.g. `schema/car/driver/first` and `schema/driver/firstName`. Note that the root element of an XML Schema is always ‘schema’, so we do not treat the root to root as a mapping. The similarity of the two Schemas is simply the structure similarity of the two roots of the trees, as $ACS(r_1, r_2)$ has been indeed computed during the Schema matching process. It is to be pointed out that, in general, there is no similarity value associated to M_{schema} .

3 Discussions and Conclusion

The system has been tested extensively using about 600 XML Schemas in total. We evaluated both matching accuracy and computational efficiency of our system. Comparisons were made with the traditional edit distance tree matching algorithm [12] and a popular XML Schema matching system COMA [1]. The results show that our new tree matching algorithm outperforms these two methods, and can be used to match larger schemas that contain hundreds of elements. A more detailed report of the ex-

periment is described in our previous paper [4]. Compared with the edit distance tree matching algorithm, our new tree matching algorithm is both faster and more accurate in terms of precision and recall. Compared with COMA, our matching system is also satisfying. Our experiments results show that, under the condition of no human interference, our matching system works better than COMA. Finally, our algorithm allows to effectively matching Schemas containing hundreds, or even thousands of nodes.

References

1. H. Do, E. Rahm: COMA A System for Flexible Combination of Schema Matching Approaches. VLDB 2002.
2. A. Doan, P. Domingos, A. Halevy: Reconciling Schemas of Disparate Data Sources: A Machine-learning Approach. In proc. SIGMOD Conference, 2001
3. A. Gupta, N. Nishimura: Finding Largest Subtrees and Smallest Supertrees. *Algorithmica*, 21:183-210, 1998
4. Jianguo Lu, Ju Wang, Shengrui Wang: An Experiment on the Matching and Reuse of XML Schemas, 5th Int. Conf. on Web Engineering (ICWE 2005), July 2005, Sydney, Accepted.
5. J. Madhavan, P. A. Bernstein, E. Rahm: Generic Schema Matching with Cupid. VLDB, 2001.
6. S. Melnik, H. Garcia-Molina, E. Rahm: Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. ICDE 2002.
7. P. Mitra, G. Wiederhold, M. Kersten: A Graph-oriented Model for Articulation of Ontology Interdependencies. EDBT 2000, LNCS Vol.1777, Springer Verlag, 2000, 86-100.
8. E. Rahm, P. A. Bernstein: A Survey of Approaches to Automatic Schema Matching. VLDB J., 10(4):334-350, 2001.
9. T. Schlieder, F. Naumann: Approximate Tree Embedding for Querying XML Data, ACM SIGIR 2000 Workshop On XML and Information Retrieval, Athens, Greece, July 28, 2000.
10. D. Shasha, J. Wang, K. Zhang, F. Y. Shih: Exact and Approximate Algorithms for Unordered Tree Matching. *IEEE Trans. on Sys., Man, and Cyber.* Vol. 24, NO.4, April 1994.
11. H. Su, S. Padmanabhan, M. Lo: Identification of Syntactically Similar DTD Elements for Schema Matching. WAIM, 2001.
12. J. Wang, B. A. Shapiro, D. Shasha, K. Zhang, K. Currey: An Algo. for Finding the Largest Approx. Common Substructures of Two Trees. *IEEE Trans. PAMI* 20, 1998, 889-895.
13. J. T. Yao M. Zhang: A Fast Tree Pattern Matching Algorithm for XML Query. Proc. of the IEEE/WIC/ACM Int. Conf. on Web Intelligence, Beijing, Sept 20-24, 2004, pp235-241.

Bi-directional Ontology Versioning BOV

Siyang Zhao and Brendan Tierney

School of Computing, Dublin Institute of Technology,
Kevin Street, Dublin 2, Republic of Ireland
Siyang.Zhao@comp.dit.ie, Brendan.Tierney@dit.ie

Abstract. This paper defines a new type of ontology versioning: Bi-directional Ontology Versioning: BOV. BOV provides bi-directional mappings and transformations between concepts in two ontology versions. BOV is identified by two levels mapping processes: linguistic mapping and structural mapping. BOV can satisfy the requirement of mapping in distributed environment.

Keywords: BOV, Bi-directional.

1 Introduction

Ontologies are increasing in popularity and show their importance in many fields such as knowledge engineering [1], knowledge representation [2], information integration [3], and etc. With the increasing uses of ontologies, a serious problem emerges: ontology changes and evolutions. Domain changes, adaptations of different tasks, or changes in the conceptualizations might cause modifications of the ontologies. Each change may create a new ontology version, which will cause many versions existing in the web. This may cause incompatibility between the data sources that use different ontology versions and give incorrect interpretations of data or make data inaccessible.

Therefore, ontology versioning is needed to handle the changes and the evolutions of ontologies. Ontology versioning is defined as “*the ability to handle changes in ontologies by creating and managing different variants of it*” [4]. This paper uses ontology versioning to describe the mappings between ontology versions.

Many data sharing and data reuse processes among distributed resources are bi-directional processes, which require bi-directional correspondences between ontology versions which describe the data of them. The existing ontology versioning approaches can only provide single-directional mappings between ontology versions, such as Ontoview [5] and PROMPTDiff [6]. When single-directional versioning approaches are used for bi-directional data sharing, they create two separated inverse single-directional mappings. This is time and effort consuming, because in these two separated inverse single-directional mapping processes, a lot of work is repeated, such as identifying mapping elements in different ontology versions. So a bi-directional ontology versioning is required for bi-directional data sharing and reuse.

Bi-directional Ontology Versioning (BOV) can relate the elements in different ontology versions in both directions at the same time and provide bi-directional transformation between them [7]. The major advantage of the bi-directional mapping is

that less time and effort are required for mapping identification and transformation in bi-directions.

2 Bi-directional Ontology Versioning (BOV)

The data sharing and data reuse processes are bi-directional processes. The single-directional ontology versioning and mapping are time-consumed and effort-consumed and can not fully satisfy the requirement of data sharing and data reuse. This paper has defined a new type of ontology versioning: Bi-directional Ontology Versioning BOV. BOV can improve the ability of data sharing and data reuse between different data resources that use different versions of the same ontology.

BOV contains four components: mapping elements, bi-directional mapping relation, bi-directional transformation and metadata. Figure 1 shows the structure of BOV.

Mapping Element			Mapping Element		
Ontology Version Title	Hierarchy	Label	Ontology Version Title	Hierarchy	Label
Bi-directional Mapping Relation					
Bi-directional Transformation					
Metadata					

Fig. 1. Bi-directional Ontology Versioning (BOV)

Definition 1. Bi-directional Ontology Versioning (BOV)

The bi-directional ontology versioning can be formalized by the 5-tuple $\langle E_1, E_2, R, T, M \rangle$

- E_1 and E_2 : two mapping elements from different ontology versions. The mapping elements contain the label and the hierarchy of the element and ontology version's title.
- R : the bi-directional mapping relation between the mapping elements, whose value is an element of the set $\{Equivalence, Isomorphic, Isomerous, Multiple_change\}$.
- T : the bi-directional transformation, which transforms the mapping elements between each other in both directions.
- M : the metadata about mappings.

2.1 Mapping Elements

The mapping elements indicate what elements from two ontology versions can be related and transformed. Figure 1 shows that the mapping element in BOV includes two elements from different ontology versions. These two elements have the same structure. The mapping element in BOV has three parts:

- **Ontology version title:** it describes the version title of the ontology version that the mapping element belongs to. It is used to identify the ontology version.
- **Hierarchy:** it includes all the ancestors of the mapping elements. Hierarchy is used to identify the mapping element in ontology versions.
- **Label:** label of the mapping element is used to describe the name of the mapping element and identify the mapping element.

2.2 Bi-directional Mapping Relation

Four bi-directional mapping relations are defined to describe the relationship between mapping elements: Equivalence, Isomorphic, Isomerous and Multiple_Change.

Definition 2. Bi-directional mapping relations

Suppose that there exist two ontology versions $V1$ and $V2$. Suppose also that $E1$ is an element in V and $E2$ is an element $V2$. $L(\text{element})$ expresses the label of an element. $S(\text{element}, \text{version})$ is the set that includes all the directly related elements and all the ancestors of the mapping element in ontology version.

- *Equivalence: When $L(E1)=L(E2)$ and $S(E1, V1)=S(E2, V2)$, the relationship between $E1$ and $E2$ is “Equivalence”.*
- *Isomorphic: When $L(E1) \neq L(E2)$ and $S(E1, V1)=S(E2, V2)$, the relation between $E1$ and $E2$ is “Isomorphic”. The uses of the synonyms and different name representations would produce the “Isomorphic” relation between elements in two versions*
- *Isomerous: When $L(E1)=L(E2)$ and $S(E1, V1) \neq S(E2, V2)$, the relation between $E1$ and $E2$ is “Isomerous”. A “Isomerous” relation would involve addition or deletion of a concept, changes to the properties of a concept, etc.*
- *Multiple_change: When $L(E1) \neq L(E2)$ and $S(E1, V1) \neq S(E2, V2)$, the relation between $E1$ and $E2$ is “multiple_change”. It means that there are linguistic changes and structure changes between $E1$ and $E2$.*

The set of bi-directional mapping relations is complete and non-overlapped. Complete here means they can describe all the relationship between the mapping elements. Non-overlapped here means the mapping cases they describe won't be overlapped, that is, one mapping case has and only has one bi-directional mapping relation for it.

2.3 Bi-directional Transformation

Transformation between elements is very important for the data sharing and reusing. It is only by transformation that ontology versions can understand and efficiently share the data of each other. The bi-directional transformation can help the data resources that use different ontology versions understand each other and freely share and exchange data.

The bi-directional transformation expression contains a bi-directional transformation expression. The transformation expression describes the transformation between the mapping elements. Usually the bi-directional transformation only supports 1:1 transformation if there is no transformation metadata supplied, because the system can not automatically extract two variants from one without any addition information.

2.4 Metadata

Metadata shows some additional information about the elements and change of ontologies such as data, author, and purpose of the change of versions. It is important for some data-sharing situations. For example, if there are two telephone companies want to share the data of the call tariff between each other, then the metadata to describe the call time is very important, because different time has different price of calls. The metadata would be supplied by the domain expert and the user, such as time and date, background and the task-dependent transformation metadata.

3 Mapping Identification

Two levels mappings are used to identify BOV: linguistic mapping and structural mapping. Linguistic mapping is used to compare the labels of elements and compute the linguistic similarity between elements. The result of the linguistic mapping is a linguistic similarity matrix. The algorithm is based on the *edit distance*, which is described in [8]. The *edit distance* measures the minimum number of token insertions, deletions, and substitutions required to transform one string into another using a dynamic programming algorithm. The Linguistic Similarity LS is computed as:

Suppose there are two sets of the names \mathcal{L}_1 and \mathcal{L}_2 of the elements in two different ontology versions V_1 and V_2 . Suppose also that there are two names of elements $L_i \in \mathcal{L}_1$ and $L_j \in \mathcal{L}_2$. The Linguistic Similarity between L_1 and L_2 is:

$$LS(L_i, L_j) := \max\left(0, \frac{\min(|L_i|, |L_j|) - ed(L_i, L_j)}{\max(|L_i|, |L_j|)}\right) \in [0, 1].$$

LS returns a degree of

similarity between 0 and 1, where 1 means perfect map and 0 means worst map.

The structural mapping compares the structures of the elements in the taxonomies of different ontology versions and computes the structural similarities between them.

Suppose that there are two elements E_1 and E_2 from two different hierarchies H_1 and H_2 in different ontology versions. The set $N(E_i, H)$ includes all the neighbour elements of E_i in H .

$N(E_i, H) := \{E_j \in E \mid H(E_j, E_i) \vee H(E_i, E_j)\}$. The set $Anc(E_i, H)$ includes all the ancestor elements of E_i in H .

The Structure Similarity (SS) between H_1 and H_2 as seen from the nodes may then be computed by following.

$$SS(E_1, H_1, E_2, H_2) = \frac{|(N(E_1, H_1) \cup Anc(E_1, H_1)) \cap (N(E_2, H_2) \cup Anc(E_2, H_2))|}{|(N(E_1, H_1) \cup Anc(E_1, H_1)) \cup (N(E_2, H_2) \cup Anc(E_2, H_2))|}$$

These mapping algorithms are based on Maedche's algorithm [9] and do some modifications to fit for ontology versioning.

From LS matrix and SS matrix, the weighted similarity WS is computed. The weighted similarity (WS) is a mean of LS and SS : $WS = w \times SS + (1-w) \times LS$, where w is the weight value in the range 0 to 1. Mapping elements are generated using the computed linguistic and structural similarities. A threshold th_{accept} can be set to identify the mappings. If $(WS(s, t) \geq th_{accept})$, then a mapping element from s to t is

turned. The values of the weighted value w and the threshold th_{accept} can be decided by the user or suggested by the system.

4 Related Work

Ontoview is a web-based system that provides support for the versioning of online ontologies [5]. Ontoview classifies different types of differences by highlighting them in different colours. PROMPTDiff compares different ontology versions and relate similar elements of them [6]. PROMPTDiff uses a set of heuristic matchers, each of which can deal with a specific mapping occasion and a fixed-point algorithm to combine the results of the matchers to produce a structural mapping between two versions. MRAFA is an Ontology Mapping FRAmework (MAFRA) for distributed ontologies in the semantic web [9]. MAFRA uses semantic bridges to generate mappings between ontologies elements. ONION (ONtology compositiON) system is an architecture based on a sound formalism to support a scalable framework for ontology integration [10]. ONION uses articulations of ontologies to interoperate among ontologies and uses articulation ontology to describe the mappings between two ontologies. GLUE is an ontology mapping system that uses machine-learning techniques to semi-automatically create semantic mappings between ontologies [11]. It computes the similarities between concepts in different ontologies based on a multi-strategy learning approach. KRAFT is an agent architecture for the integration of heterogeneous information systems [12]. KRAFT uses “ontology clustering” to integrate the heterogeneous ontologies. Table 1 describes the comparison of the features of these mapping approaches.

Table 1. Comparison of Single-directional Mapping Approaches

	Ontoview	PromptDiff	MAFRA	ONION	GLUE	KRAFT
Ontologies Mapping			√	√	√	√
Ontology Versioning	√	√				
M:N mapping			√	√		√
Linguistic level Mapping	√	√	√	√	√	√
Structure level Mapping			√	√	√	
Similarity computation			√	√	√	
Automatic		√				
Semi-automatic	√		√	√	√	√

Ontology mapping maps different ontologies and ontology versioning maps different ontology versions. In these approaches, only MAFRA, ONION and KRAFT can generate m: n mappings. The linguistic mapping identifies similarities between labels of the elements of the different ontologies or ontology versions. The structure mapping identifies the similarity between the taxonomies of elements in different ontologies or ontology versions. Similarity-based technique uses some matching algorithms

to compute the similarity between elements from ontological resources and identify the mappings based on the similarities. The rule-based technique uses some rules and constraints to identify the mappings. Most of these approaches require some metadata for generating the mapping that are provided by the user, which means that they are all semi-automatic. There is only one approach PromptDiff that is fully automatic.

5 Conclusion

This paper introduced a new type of ontology versioning: bi-directional ontology version (BOV), which can map two ontology versions bi-directionally at the same time. This paper gave the formal definition of BOV and described the components of it. The innovation of BOV lies in the bi-directional character. BOV can relate and transform similar elements of different ontology versions in both directions. This will improve the ability of data sharing and reuse among distributed data resources.

There are still some limitations of BOV. When there are many synonyms and homonyms, the accuracy of the mapping will be affected. The mapping algorithms can not identify the synonyms and homonyms by themselves alone. Some other technologies such as WordNet can help to solve this problem. BOV only supports 1:1 mapping if there is no transformation metadata provided by the user. So some single-directional only multiple candidates mapping will be lost in the bi-directional mapping. And that would induce losing some related knowledge between the elements in different ontology versions.

References

1. Gruber, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition* (1993).
2. Guarino, N.: Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human and Computer Studies*, 43(5-6): 625-640 (1995)
3. Gray, P.M.D., Preece, A., Fiddian, N. J., Gray, W.A., Bench-Capon, T.J.M., Shave, M.J.R., Azami, N., and Wiegand, M.: KRAFT: Knowledge fusion from distributed databases and knowledge bases. In R.R. Wagner, editor, *Eighth International Workshop on Database and Expert System Applications (DEXA-97)*, pages 682—691 (1997).
4. Klein, M. and Fensel, D.: Ontology versioning for the Semantic Web. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, pages 75-91, Stanford University, California, USA, July~30 - August~1 (2001).
5. Klein, M., Fensel, D., Kiryakov, A. and Ognyanov, D.: Ontoview: Comparing and versioning ontologies. In *Collected Posters ISWC 2002*, Sardinia, Italy (2002).
6. Noy, N. F and Musen, M. A.: PromptDiff: A Fixed-Point Algorithm for Comparing Ontology Versions. In *the Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, Edmonton, Alberta (2002)
7. Zhao, Siyang and Tierney, Brendan: Bi-Directional Mapping between Ontology Versions: A Requirement for Data Sharing in Distributed Environments, *Second International Conference on Computer Science and its Applications (ICCSA-2004)*, San Diego, California, USA, June 28-30, (2004).

8. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.*, (1966).
9. Maedche, A., Motik, B., Silva, N., Volz, R: MAFRA—A Mapping FRAMework for distributed ontologies. *Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW*, Madrid, Spain, (2002).
10. Mitra, P., Wiederhold, G., and Kersten, M.: A Graph-Oriented Model for Articulation of Ontology Interdependencies. In *Proceedings Conference on Extending Database Technology 2000 (EDBT'2000)*, Konstanz, Germany (2000).
11. Doan, A. H., Madhavan, J., Donfingos, P., Halevy, A. Y. :Learning to Map between Ontologies on the Semantic Web. In *Proc. of the Int. WWW Conference*, (2002).
12. Gray, P.M.D., Preece, A., Fiddian, N. J., Gray, W.A., Bench-Capon, T.J.M., Shave, M.J.R., Azarmi, N., and Wiegand, M.: KRAFT: Knowledge fusion from distributed databases and know ledge bases. In R.R. Wagner, editor, *Eighth International Workshop on Database and Expert System Applications (DEXA-97)*, pages 682—691 (1997).

Quantitative Modeling for Web Objects' Cacheability*

Chen Ding¹, Chi-Hung Chi², Lin Liu², LuWei Zhang³, and H.G. Wang³

¹School of Computer Science, Ryerson University, Canada

²School of Software, Tsinghua University, Beijing, China

³School of Computing, National University of Singapore, Singapore
chichihung@mail.tsinghua.edu.cn

Abstract. In this paper, we propose a mathematical model, called the **E-Cacheability Index**, to quantify and compare the effectiveness of caching web objects on Internet. Unlike previous related work on web object cacheability, our study is based on a comprehensive study of all object attributes related to the HTTP protocol transfer, content reuse, freshness re-validation as well as the inter-relationship/dependency among these attributes in the final caching decision. This model is important because it not only estimates the cacheability effectiveness of web content, but also provides hints on web caching optimization.

1 Introduction

There has been a lot of research work being carried out on web caching methods and architectures [7]. There are also works focusing on the acceleration of web accesses through techniques like prefetching, pre-connection, and encoding/compression. However, little work is found on studying factors that affect the effectiveness of caching web objects. By "effectiveness", we refer to the longest time an object in cache is considered as "fresh" or "validation but without actual data transfer". To gain an in-depth understanding to this effectiveness measure, we not only need to understand all factors that actually cause an object to be cacheable or non-cacheable, but we also have to analyze the inter-relationship and dependency among them. Furthermore, we would also like to quantify the effectiveness of caching an object from the viewpoint of the intrinsic attributes of web content. Only after all these are found then we know exactly how to fix or optimize the attribute setting (e.g. HTTP header fields) of web content and caching management (e.g. replacement) for better content reuse.

This paper attempts to provide an in-depth analysis on how various network factors and content attributes contribute to the final decision of web caching. While previous research work on web object cacheability often focuses on non-cacheable objects, we would like to study both cacheable and non-cacheable objects so that a more complete understanding can be obtained. To achieve this goal, a formal mathematical model, called the **E-Cacheability Index**, is proposed to express the effectiveness of caching an object using a numerical index. The basic idea behind this model is to provide

* This research is supported by the funding 2004CB719400 of China.

precise description on the "availability", "freshness", and "validation" of web content if it is stored in local web caches. With this index, not only can the cacheability of web objects be quantified, but it also provides a mechanism to compare the relative cacheability effectiveness among web objects.

2 Related Work

As one of the earliest studies on web caching, Harvest system [2] faced the difficulty to identify non-cacheable objects. Its intermediate solution was to scan the URL name to detect CGI scripts. It also discarded large cacheable objects over predefined size limit. Their implementation was popular at the early stage of web [6]. The impacts of caching-related HTTP headers on cacheability decision were investigated in several trace-base studies. [4] collected network traces for modem traffic at UCB in 1996 and analyzed the cacheability of requested web objects. In their study, many caching-related HTTP header fields were examined such as "Pragma: no-cache", "Cache-control", "Expires" and "Last-Modified". Ignoring cookies, their experiment results showed that non-cacheable responses were quite low, so did the CGI response. [1] noticed the bias in [4] and considered cookies in their experiments. Their results showed that non-cacheable requests due to cookies could be up to 30%. Later studies on different traces in [3] and [9] showed that the overall rate of non-cacheability was around 40%. There are other studies on cacheability based on active content monitoring [8]. More complete analysis on content non-cacheability can be found in [11][5][10]. [11] concluded that the main reasons for non-cacheability included responses from server scripts, responses with cookies and responses without Last-Modified header. [5] proposed a complicated method to classify content cacheability using neural network. More recently, [10] analyzed the combinational effects of the causes in content non-cacheability and proposed quantity measure to characterize object cacheability.

3 E-Cacheability Index

The final decision on the cacheability of an object is actually made in the caching proxies. Apart from obeying the HTTP protocol directives, caching proxies also have their own preferences to determine whether they should cache the object according to their own architecture and policies. Thus, to better understand an object's cacheability, we need to analyze the combinational effects of relevant content settings on the effectiveness of caching an object. For this purpose, our method employs an index, called the *E-Cacheability Index* (Effective Cacheability Index), which is a relative numerical value derived from our proposed formal model of object cacheability.

3.1 Basic Concepts

Based on the basic working mechanism of a proxy, there are three attributes that determine an object's *E-Cacheability Index*. They are object availability to be cached, data freshness and validation frequency. Their relationship is shown in the equation (Equation (1)) below.

$$E\text{-Cacheability Index} = \text{Availability_Ind} * (\text{Freshness_Ind} + \text{Validation_Ind}) \quad (1)$$

Unlike previous study on object cacheability, which just determines if an object can be cached, *E-Cacheability Index* goes one step further. It also measures the effectiveness of caching an object by studying the combinational effect of the three factors of caching availability, data freshness, and validation frequency.

In the equation above, the *Availability_Ind* of an object is used to indicate if the object is available for caching or not. If the object is not available, the *E-Cacheability Index* of the object will be zero. Thus, all non-cacheable objects have an *E-Cacheability Index* of zero, and under this case, the meaning of the other terms (*Freshness_Ind* and *Validation_Ind*) is undefined. Hence, *Availability_Ind* is in the most dominant position in our measurement. After the indication of whether the object is cacheable from the *Availability_Ind* attribute, the *Freshness_Ind* and *Validation_Ind* attributes are then important to measure how effective caching this object is. *Freshness_Ind* is a period that indicates the duration of data freshness of an object, and *Validation_Ind* is an index that indicates the probability of the staleness of an object, using the frequency of the need to revalidate the object.

The *E-Cacheability Index* is defined by these two attributes once an object is determined to be available for caching. The longer the period of data freshness and the lower the frequency of re-validation result in a higher *E-Cacheability Index*. Thus, larger value of the *E-Cacheability Index* indicates higher potentials to cache this object. Furthermore, for objects with smaller *E-Cacheability Index*, detailed analysis can give hints on which content settings have larger influence on the effective cacheability of an object. This can help to optimize the content settings for better caching. In Equation (1), the "*" operator is used to handle the situation when an object is non-cacheable. As will be seen in later sections, it will enforce the resulting index to be zero for non-cacheable objects. The "+" operator is used to separate the two situations of reusing the cached content by shifting the index to two exclusive regions – the region of negative values to indicate the need for revalidation each time an object is used, and the region of value greater than or equal to one to give a quantitative measure on the caching effectiveness.

In the next few sections, we will describe, based on the actual request methods, response codes, header fields, and proxy, the detailed composition of each term in the equation above. We will use *I* in the equations to indicate request information, and *O* to indicate response information.

3.2 Availability_Ind

The term *Availability_Ind* in Equation (1) is defined as the overall composition of all factors that will possibly affect the caching availability of an object. The possible

value of this term is 0 (non-cacheable) or 1 (cacheable). Its *Availability_Ind* equation is shown below (in Equation (2)):

$$Availability_Ind = I_{RM(A)} * O_{SC(A)} * O_{HD(A)} * O_{pp(A)} * I_{HD(A)} \quad (2)$$

where $I_{RM(A)}$ refers to the request method sent, $O_{SC(A)}$ refers to the response code related to object availability, $O_{HD(A)}$ refers to the header fields in the response that influence the availability of cacheability of the object, $O_{pp(A)}$ refers to the proxy preference in the response, and $I_{HD(A)}$ refers to the relevant header fields that influence availability in the request. The value of *Availability_Ind* is either zero (non-cacheable) or one (cacheable). Equation (2) uses the associative operator (*), signifying that an object is non-cacheable (not available for caching) if there exists at least one factor that suggests the non-availability of the object in cache.

3.3 Freshness_Ind

The term *Freshness_Ind* in Equation (1) is defined as the overall composition of all factors that will possibly affect the data freshness of an object. The possible value of this term is zero for non-cacheable object to value greater than zero for cacheable objects. Its equation is shown below (in Equation (3)):

$$Freshness_Ind = I_{RM(F)} * O_{SC(F)} * O_{HD(F)} \quad (3)$$

where $I_{RM(F)}$ refers to the request method sent, $O_{SC(F)}$ refers to the response code related to data freshness, and $O_{HD(F)}$ refers to the relevant header fields that influence the data freshness in the response. In Equation (3), the associative operator (*) indicates that a non-cacheable response will result in the entire equation to be zero ($I_{RM(F)}$ and $O_{SC(F)}$). Otherwise, the *Freshness_Ind* value of the object will be determined by the relevant header fields in the response ($O_{HD(F)}$).

3.4 Validation_Ind

The term *Validation_Ind* in Equation (1) is defined as the overall composition of all factors that will possibly affect how valuable an object is in terms of its validation requirement. The possible values of this term is 0 (non-cacheable), -1 (if object must be revalidated each time even though it is cacheable), and greater than 1 (if object is cacheable). Its *Validation_Ind* equation is shown below (in Equation (4)):

$$Validation_Ind = I_{RM(V)} * O_{SC(V)} * OR_val-op(I_{HD(V)}, I_{pp(V)}, O_{HD(V)}) \quad (4)$$

where $I_{RM(V)}$ refers to the request method, $O_{SC(V)}$ refers to the status code, $I_{HD(V)}$ refers to the relevant header fields in the request that influence validation, $O_{HD(V)}$ refers to the relevant header fields in the response that influence validation, and $I_{pp(V)}$ refers to the proxy preferences in the request that influence validation. For function $OR_val-op(a_1, \dots, a_n)$, where $a_i \in \{-1, 0, 1\}$, its value is as follows:

$$OR_val-op(a_1, \dots, a_n) = \begin{cases} -1 & \text{there exists at least one } a_i \text{ with value } -1 \\ 1 & \text{there exists at least one } a_i \text{ with value } 1 \text{ \& no } a_i \\ & \text{with value } -1 \\ 0 & \text{all } a_i \text{ with value } 0 \end{cases}$$

Equation (4) indicates that a non-cacheable response will result in the equation being zero ($I_{RM(V)}$, $O_{SC(V)}$). Otherwise, the value in the equation will either be 1 or -1 depending on the input parameters of the OR_val-op operator.

3.5 E-Cacheability Index

Based on Equation (1) and substituting the terms of all factor equations in (2), (3) and (4) into it, we have (in the rest of the paper, we will use “ EC ” to shorten “ E -Cacheability Index”):

$$EC = I_{RM(A)} * O_{SC(A)} * O_{HD(A)} * O_{pp(A)} * I_{HD(A)} * (I_{RM(F)} * O_{SC(F)} * O_{HD(F)} + I_{RM(V)} * O_{SC(V)} * OR_val-op(I_{HD(V)}, I_{pp(V)}, O_{HD(V)}))$$

Since the request term in *Availability_Ind*, *Freshness_Ind* and *Validation_Ind* equations must be the same and is defined for the same object, $I_{RM(A)} = I_{RM(F)} = I_{RM(V)} = I_{RM}$. Following the same argument, the status code of the response is the same response since all three factors are defined for the same object. Hence, let $O_{SC(A)} = O_{SC(F)} = O_{SC(V)} = O_{SC}$. Then,

$$\begin{aligned} EC &= I_{RM} * O_{SC} * O_{HD(A)} * O_{pp(A)} * I_{HD(A)} * (I_{RM} * O_{SC} * O_{HD(F)} + \\ &I_{RM} * O_{SC} * OR_val-op(I_{HD(V)}, I_{pp(V)}, O_{HD(V)})) \\ &= I_{RM}^2 * O_{SC}^2 * O_{HD(A)} * O_{pp(A)} * I_{HD(A)} * (O_{HD(F)} + OR_val-op(I_{HD(V)}, I_{pp(V)}, O_{HD(V)})) \end{aligned} \quad (5)$$

The value of I_{RM} and O_{SC} can be easily determined:

$$\begin{aligned} I_{RM} &= \begin{cases} 1 & \text{if method is GET, POST or HEAD} \\ 0 & \text{otherwise} \end{cases} \\ O_{SC} &= \begin{cases} 1 & \text{if status code is 200, 203, 206, 300, 301, 410} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Given the values of I_{RM} and O_{SC} above, Equation (5) can be simplified as:

$$EC = I_{RM} * O_{SC} * O_{HD(A)} * O_{pp(A)} * I_{HD(A)} * (O_{HD(F)} + OR_val-op(I_{HD(V)}, I_{pp(V)}, O_{HD(V)})) \quad (6)$$

This equation is the final formula to compute the effectiveness of caching an object.

4 Conclusion

Despite of the numerous ongoing research efforts in web caching, most of them concentrate on whether an object should be cached. There is no further analysis on the cacheability of a cached object. The proposed Effective Cacheability (*E-Cacheability Index*) model presented in this paper attempts to go one step further, by (i) first determining whether an object can be cached, and (ii) further determining the effectiveness of caching such an object, if it is cached. This is done in the form of a relative numerical value, which can be used as a quantitative measurement for the effectiveness of caching the object.

References

- [1] R. Cáceres, F. Douglis, A. Feldmann, G. Glass, and M. Rabinovich, "Web Proxy Caching: the Devil is in the Details", *Proceedings of the 1998 Workshop on Internet Server Performance*, Madison, WI, June 1998.
- [2] A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, K. J. Worrell, "A Hierarchical Internet Object Cache", *Proceedings of the USENIX Technical Conference*, San Diego, CA, USA 1996.
- [3] A. Feldmann, R. Cáceres, F. Douglis, G. Glass, and M. Rabinovich, "Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments", *Proceedings of IEEE Infocom'99*, March, 1999, pp. 107-116.
- [4] S. D. Gribble, E. A. Brewer, "System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace", *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, 1997.
- [5] T. Koskela, J. Heikkonen, K. Kaski, "Modeling the Cacheability of HTML Documents", *Proceedings of the 9th World Wide Web Conference*, 2000.
- [6] S. Manley, M. Seltzer, "Web Facts and Fantasy", *Proceedings of the 1997 USENIX Symposium on Internet Technologies and Systems*, December 1997.
- [7] D. Wessels, *Information Resource Caching FAQ*.
URL: <http://ircache.nlanr.net/Cache/FAQ/>
- [8] C. E. Wills, M. Mikhailov, "Towards a Better Understanding of Web Resources and Server Responses for Improved Caching", *Proceedings of the 8th International World Wide Web Conference*, Toronto, Canada, May 1999.
- [9] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. Karlin, H. M. Levy., "On the Scale and Performance of Cooperative Web Proxy Caching", *Proceedings of the Seventeenth ACM Symposium on Operating Systems Principles*, Kiawah Island, SC, December 1999.
- [10] L. W. Zhang, "Effective Cacheability", Internal Technical Report, National University of Singapore, 2002.
- [11] X. Zhang, "Cacheability of Web Objects", *Master Thesis of Computer Science Department, Boston University*, USA, 2000.

An Efficient Scheme of Merging Multiple Public Key Infrastructures in ERP[★]

Heng Pan¹, YueFei Zhu¹, ZhengYun Pan², and XianLing Lu²

¹Institute of Information Engineering, Information Engineering University,
Zhengzhou, China 450002

²Institute of Electronic Technology, Information Engineering University,
Zhengzhou, China 450004
panhengpan@hotmail.com

Abstract. The *Public Key Infrastructure* (PKI) can provide various secure services in E-commerce. Once the enterprise changes its collaborators, the multiple PKIs deployed by the enterprise and its new collaborators must be interoperated. Such things often happen especially in the enterprises adopting *Enterprise Resource Planning* (ERP) system. Considering the specialties of such context, an efficient scheme based on hierarchy structure is proposed. Keeping the previous PKIs integrate and independent, the merging process is quick and low-cost. Furthermore, compared with the common ways like mesh certificate authority model and Bridge certificate authority model, the certificate verification in new PKI is more efficient. In order to precisely describe the scheme, some essential conceptions of PKI are formalized.

1 Introduction

With the development of E-commerce, many enterprises have built their own Public Key Infrastructures (PKIs) to support the various web applications [2]. Based on web technology, ERP (Enterprise Resource Planning) can promptly reflect the changes of the market requirements and help the enterprise to select the appropriate collaborators to do transactions [3]. To guarantee the security of the business information, the corresponding PKIs of the enterprise and the collaborators should be dynamically inter-operated.

Such PKI interoperability problem takes special characteristics. First, the interoperability among these PKIs is temporary, which will dynamically change with the market requirements. Thus, the merging process needs to be low-cost, easily constructed and flexible. Secondly, the enterprise who selects the collaborators plays as principal and the collaborators play as subordinate. Therefore, the corresponding PKIs also have the “principal and subordinate” relationships.

[★] This work has been supported in part by the National Natural Science Foundation of China (90204015, 60473021), the Basic Research Program of China (973 Program) (G1999035804) and the Science Foundation of Henan Province (511010900).

According to these specialties, this paper proposes an efficient scheme. Based on the hierarchy structure, our scheme can quickly merge the PKIs of the enterprise and its collaborators. Issuing few certificates, the new PKI can turn from the existing PKIs smoothly and quickly. Furthermore, with simple and efficient certificate verification, the new PKI insures the secure transactions between the enterprise and its collaborators.

The rest of the paper is organized as follows. In section2, first, the definitions of some conceptions in PKI are formalized. Then, our proposed scheme and the certificate processing are described in detail based on these formal definitions. Section 3 analyzes the scheme performance. Finally, our future work is discussed in the section of conclusion.

2 Proposed Scheme

Our proposed scheme is based on the following assumptions: According to the market requirements reflected by ERP, the enterprise E^* selects n ($n \geq 1$) enterprises (E_1, E_2, \dots, E_n) as its collaborators. Suppose that the PKI of E^* (denoted by Φ^*) consists of a certificate authority CA^* and a set of users U^* ($\Phi^* = CA^* \cup U^*$). Let every PKI (denoted by Φ_i) of E_i also take one certificate authority CA_i and a set of users U_i , where $|U_i| = m$, $\Phi_i = CA_i \cup U_i$ and $1 \leq i \leq n$. Let $\bar{\Phi}$ ($\bar{\Phi} = \Phi^* \cup \Phi_1 \cup \Phi_2 \cup \dots \cup \Phi_n$) be the set of all the entities in this system. To address our scheme clearly, first, some essential conceptions of PKI are formalized.

2.1 Definition

Definition 1. A certificate takes the following form: $Cert_i^s = Cert(I, S, D, pk_s, Sig_i, A)$, where I is the issuer, S is the subject of the certificate, pk_s is the public key of S , D is the validity period of the certificate, Sig_i is the signature of the issuer I , and A represents the additional data.

Definition 2. Let \downarrow be a trust relationship over set $\bar{\Phi}$, where $a \downarrow b = \{(a, b) \mid a, b \in \bar{\Phi}, \exists Cert_a^b = cert(a, b, D, pk_b, Sig_a, A)\}$.

The trust relationship \downarrow has some properties: (1) If $a \downarrow b$ then b trusts a , b takes the certificates issued by a . (2) The trust relationship \downarrow can be transferred. If $a \downarrow b, b \downarrow c$, then $a \downarrow c$.

2.2 The Merging Process

Our proposed scheme is to quickly merge the Φ^* and Φ_i into a new hierarchical PKI system. Based on the assumptions, it is obvious that the enterprise E^* plays as principal and the collaborator E_i play as subordinate. Therefore, CA^* will be chosen as the root certificate authority of $\bar{\Phi}$, CA_i will become the subordinate certificate authority (see Fig.1).

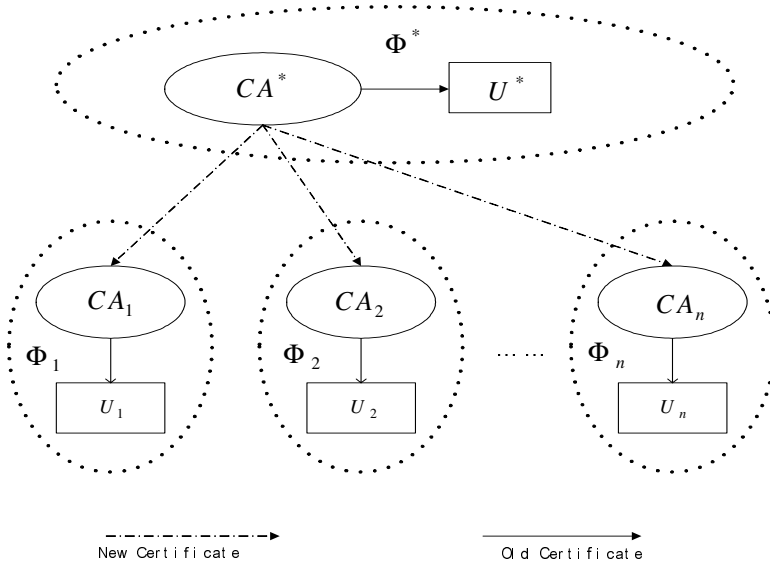


Fig. 1. The PKIs Merging Model

In the merging process, first, CA^* should be trusted by all the entities of $\bar{\Phi}$. In other words, the certificate of CA^* must be securely stored by every entities. Then, to complete the whole merging process, CA^* need to issue the new certificates to the entities. For E_i , not only collaborates with E^* , but may collaborate with other enterprises as well, the scheme needs keep Φ_i as integrate and independent as possible. Our scheme attains that by only issuing the new certificate to CA_i and reserving the structure and the users' certificates of Φ_i .

No matter which collaborator it is, the merging process is the same. Thus, we take merging Φ^* and Φ_n as an example to illustrate the whole process. The main steps are as follows:

- (1). CA^* distributes its certificate $Cert_{CA^*}^{CA^*} = Cert(CA^*, CA^* D, pk_{CA^*}, Sig_{CA^*}, A)$ to CA_n .

$$CA^* : CA_n \leftarrow Distribute(Cert_{CA^*}^{CA^*})$$

- (2). CA_n issues a temporary certificate $Cert_{CA_n}^{CA^*} = Cert(CA_n, CA^*, D, pk_{CA^*}, Sig_{CA_n}, A)$ to CA^* . Then, CA_n broadcasts both $Cert_{CA_n}^{CA^*}$ and $Cert_{CA^*}^{CA^*}$ to all its users in set U_n .

$$CA_n : Store(Cert_{CA^*}^{CA^*});$$

$$CA^* \leftarrow Issue(Cert_{CA_n}^{CA^*});$$

$$U_n \leftarrow Broadcast(Cert_{CA^*}^{CA^*}, Cert_{CA_n}^{CA^*})$$

- (3). As soon as getting $Cert_{CA_n}^{CA^*}$, the user in U_n verifies its signature by using the public key pk_{CA_n} of CA_n . If the signature is true, the user can acquire CA^* 's public key pk_{CA^*} .

Then, the user verifies the signature of $Cert_{CA^*}^{CA^*}$ using pk_{CA^*} . If it is true, the user will store $Cert_{CA^*}^{CA^*}$ and trust the new root certificate authority CA^* .

```

 $u_j (u_j \in U_n)$ :
    for  $j=1$  to  $m$ 
    do {
        if (Verify $pk_{CA_n}$  (  $Cert_{CA_n}^{CA^*}$  ) )
        then
            {
                Store (  $pk_{CA^*}$  );
                if (Verify $pk_{CA^*}$  (  $Cert_{CA^*}^{CA^*}$  ) ) then (Store(  $Cert_{CA^*}^{CA^*}$  ) )
                else Exit
            }
        else Exit
    }

```

(4). CA_n sends the message maintaining its previous public key pk_{CA_n} to CA^* and applies for its new certificate.

$$CA_n : CA^* \leftarrow Apply(pk_{CA_n})$$

(5). CA^* issues a new certificate $Cert_{CA^*}^{CA_n} = (CA^*, CA_n, D, pk_{CA_n}, Sig_{CA^*}, A)$ to CA_n . The users in set U_n still use their previous certificates, which are issued by CA_n ($Cert_{CA_n}^u = (CA_n, u, D, pk_u, Sig_{CA_n}, A)(u \in U_n)$). In this way, Φ_n maintains its integrity and independence as before.

$$CA^* : CA_n \leftarrow Issue(Cert_{CA^*}^{CA_n})$$

2.3 Certificate Verification in New PKI

After merging, the new PKI can provide many security services (like confidentiality, information integrity etc.) for the enterprise’s business processing. However, all these services are based on the correct certificate verification among the entities.

2.3.1 Certificate Verification Between Φ^* and Φ_i ($1 \leq i \leq n$)

The main purpose of our merging scheme is to guarantee the secure transactions between E^* and its collaborators E_i . In this part, we only discuss how the certificate verification works between Φ^* and Φ_i . Actually, verifying a certificate involves many aspects, for the sake of simplicity, here we only describe how to built the certificate verification path between the transmission participants.

Suppose $a \in U^*$ transfers its certificate $Cert_{CA^*}^a$ to $b \in U_i$ ($1 \leq i \leq n$). And b wants to verify $Cert_{CA^*}^a$. Because $Cert_{CA^*}^a$ is signed and issued by CA^* and pk_{CA^*} has already been stored by b , it is easy for b to verify $Cert_{CA^*}^a$.

Suppose $b \in U_i$ ($1 \leq i \leq n$) transfers its certificate $Cert_{CA_i}^b$ to $a \in U^*$. When a wants to verify $Cert_{CA_i}^b$, a simple certificate path can quickly be built. According to the definitions in 2.1, $CA^* \downarrow a$, $CA^* \downarrow CA_i$, $CA_i \downarrow b$ can be made. In terms of the transfer property of “ \downarrow ”, $CA^* \downarrow b$ is get. Therefore, a can verify $Cert_{CA_i}^b$.

2.3.2 Certificate Verification in Φ_i ($1 \leq i \leq n$)

As we have mentioned in 2.2, every $\text{PKI}(\Phi_i)$ maintains its structure integrity and keep certificate management independent. Thus, the certificate verification in Φ_i doesn't change. In order to conserve space, we will not describe it here in detail.

3 Performance Analysis

(1) Efficient Merging Process

A merging method that also uses the hierarchy structure is provided in paper [6]. However, in that method, a great amount of certificates need to be issued. Based on the assumptions in 2.1, if the enterprise E^* collaborates with n enterprises at the same time, using our proposed scheme to merge Φ^* and Φ_i ($1 \leq i \leq n$) only need issue n certificates. Whereas, if we choose the way proposed in [6], mn ($m \gg n$) users' certificates have to be issued.

Such few certificates issuances make the reconstruction process quick and low-cost. That particularly meets the requirements of the dynamic changes of PKIs in ERP.

(2) Efficient Certificate Verification in New PKI

Till now, the mesh CA model and the Bridge CA model [1][4][5] are the most common ways to solve the merging issue in PKI. For these methods are based on cross-certificate, the conspicuous shortcoming is the complex certificate path construction and verification. In contrast to them, from the description in 2.2, we can see such process is simple and efficient in our scheme. Moreover, based on the same above assumptions, in mesh CA model, $n(n-1)$ cross certificates are required to be issued. In the bridge CA model, $2n$ cross certificates are needed.

4 Conclusions

In this paper, a problem of merging multiple PKI in ERP is considered. According to the characteristics, an efficient scheme based on hierarchy structure is proposed and the formal approach is also used to describe the conceptions in PKI. Compared with those common ways, the merging process and certificate verification are both efficient,

which satisfies the requirements of the dynamic changes of PKIs in ERP. In our future works, the performance analyses will be finished in formal ways.

References

1. R. Perlman. : An Overview of PKI Trust Models. *IEEE Network*, 11/12. (1999) 38-43
2. A.Nash, W.Duane, C.Joseph, and D.Brink, *PKI: Implementing and Managing E-security*, McGraw-Hill Companies, New York, (2001)
3. M. Gillmann, J.Hertel, C.G.Jung, G.Kaufmann: *Cooking the Web-ERP*, LNCS2519, Springer-Veriag. (2002) 602-617
4. S. Lloyad.: *CA-CA Interoperability*, White Paper, [http:// www. Pki.forum.org/pdfs/](http://www.Pki.forum.org/pdfs/) ,(2004)
5. Levi,A.: *Design and Performance Evaluation of the Nested Certification Scheme and Its Application in Public Key Infrastructure*, Ph.D. Thesis, Bogazici University, Dept. of Computer Engineering, (May1999).
6. S.Koga, K.Sakurai: *A Merging Method of Certification Authorities without Using Cross-Certificates*, *Proceedings of the 18th International Conference on Advanced Information Networking and Application*. AINA, IEEE Computer Society, ISBN 0-7695-2051-0/2004, Volume2,(2004),174-177

Forms-XML: Generating Form-Based User Interfaces for XML Vocabularies

Y.S. Kuo, N.C. Shih, Lendle Tseng, and Hsun-Cheng Hu

Institute of Information Science,
Academia Sinica, Taiwan
{ yskuo, ncshi, lendle, hchu}@iis.sinica.edu.tw

Abstract. The tool support for user interfaces for XML data is inadequate. Forms-XML is an interactive component invoked by applications for generating form-based user interfaces for prescribed XML vocabularies automatically.

XML has found many applications in Web-based information systems. These information systems frequently need to present XML data (documents) to their users for interaction and updating. So far, many user interfaces for XML data have been constructed from scratch for specific XML vocabularies and applications, which is time-consuming and expensive. The tool support for user interfaces for XML data is inadequate.

Forms-XML is an interactive component (ActiveX control) invoked by applications for generating user interfaces for prescribed XML vocabularies automatically [1,2,3]. Based on a given XML schema, the component generates a hierarchy of HTML forms for users to interact with and update XML data compliant with the given schema. The user interface Forms-XML generates is very simple with an abundance of guidance and hints to the user, and can be customized by user interface designers as well as developers.

In addition to Forms-XML, we have developed a visual interface design tool, the Forms-XML Designer. It supports interface designers to specify the layout and style of a user interface for a prescribed XML vocabulary while the desired working user interface can be generated immediately by invoking the Forms-XML component.

We will demonstrate the functions of Forms-XML by demonstrating the Designer. The form-based user interface Forms-XML generates and the process for customizing a user interface for an XML vocabulary will be exhibited by examples.

References

1. Y. S. Kuo, Jasper Wang, and N. C. Shih, "Handling Syntactic Constraints in a DTD-Compliant XML Editor", Proc. ACM Symp. Document Engineering, Grenoble, France, Nov. 2003.
2. Y. S. Kuo, N. C. Shih, Jasper Wang, and Lendle Tseng, "Avoiding syntactic violations in Forms-XML", Extreme Markup Languages, Montreal, Aug. 2004.
3. Y. S. Kuo, N. C. Shih, Lendle Tseng, and Hsun-Cheng Hu, "Generating Form-Based User Interfaces for XML Vocabularies", submitted for publication.

Author Index

- Agarwal, Nitin 475
Ahn, Hyong-Jin 526
An-long, Chen 700
Asadi, Saeid 91
- Bae, Haeyoung 114
Bai, Qingyuan 368
Bao, Hong 626
Bao, Yubin 785
Bok, KyoungSoo 589
Buneman, Peter 1
- Cai, Keke 852
Cao, Jian 440
Cao, Lei 440
Chang, Chung-Yi 91
Chang-an, Yuan 700
Chang-jie, Tang 700
Che, Haoyang 694
Chen, Chun 852
Chen, Furong 184
Chen, Gang 102, 221, 380, 578, 773
Chen, Haikun 45
Chen, Hong 822
Chen, H.X. 894
Chen, Jing 297
Chen, Ke 221, 380, 773
Chen, Lian 284
Chen, M.S. 894
Chen, Shi 804
Chen, Yang 618
Chen, Yanyu 651
Chen, Yu 81
Chen, Zhi-ping 713
Cheng, Chun-Tian 267
Cheng, Zang 755
Chi, Chi-Hung 257, 556, 913
Choi, Jonghwa 749
Chuan, Li 700
Cui, Juntao 816
- Dai, Yiqi 81
Deng, Qingxu 725
Deng, Shengchun 632
Deng, Zhi-Hong 138
- Diederich, Joachim 91
Ding, Chen 257, 913
Ding, Zhigang 816
Dong, Baoli 840
Dong, Jinxiang 102, 162, 221, 380,
538, 578, 773, 882
Dong, Yabo 601
Du, Xiaoyong 209
- Eo, Sanghun 114
- Fang, Yang 275
Feng, Jianhua 32, 663
Feng, Yucan 657
Fu, Peng 428
- Gang, Chen 755
Gao, Aiqiang 308
Gao, Mingxia 184
Gao, Qing 357
Ge, Pengcheng 45
Ghafoor, Memon Abdul 162, 538
Gong, Jian 56
Greco, S. 810
Gu, Jun 694
Gu, Ning 245, 816
Gu, Xinjian 840
Gu, Yu 566
- Han, Dingyi 864
Han, Hyoil 688
Hao, Zhongxiao 392
Haque, Ehtesham 475
Hassana, G.K. 804
He, Qinming 492
He, Zengyou 632
He, Zhenying 45
Heilili, NuerMaimaiti 618
Hong, Jun 368
Hou, Lishan 320
Hu, Hsun-Cheng 925
Hu, Jian-jun 700
Hu, Ming-zeng 333
Hu, Tian-Lei 221, 380
Huang, Joshua Zhexue 502

- Huang, Shen 676
 Hwang, Chong-Sun 233

 Jang, Seokkyu 114
 Jeong, Hwa-Young 798
 Jia, Huibo 761
 Jia, Yan 706
 Jin, Cheqing 779
 Jin, Hai 25, 834
 Jin, Zhi 320
 Jing, Liping 502
 Jing, Liu 275
 Jing, Peng 700
 Jinxiang, Dong 755

 Kang, KyungWoo 876
 Kang, YunHee 876
 Kim, Kwang-Hoon 526
 Kim, MyoungHo 589
 Kim, Myungkeun 114
 Kim, Namhoon 749
 Kong, Fansheng 822
 Kuo, Y.S. 925

 Lee, Jaedong 114
 Lee, Kangsun 743
 Li, Chenyang 657
 Li, Guoliang 32, 663
 Li, Haifeng 767
 Li, Hongyan 194, 846
 Li, Hu'an 638
 Li, Hui-Xian 267
 Li, Jianzhong 45, 68, 416, 719
 Li, JingFeng 670
 Li, Man 209
 Li, Meimei 194
 Li, Meng 404
 Li, Minglu 440
 Li, Qian 404
 Li, Shanping 357
 Li, Shijun 888
 Li, Xiaoguang 56
 Li, Xiaojing 102
 Li, Xin 138
 Li, Ying 440
 Li, Zengzhi 297
 Li, Zheng 275
 Liao, Yuguo 32, 663
 Lin, Chen 682
 Lin, Chenxi 172

 Lin, Huaizhong 852
 Lin, Jian 601
 Lin, Lili 858
 Lin, Mu 713
 Lin, Xin 357
 Lin, Xuemin 68, 150
 Lin, Ya-ping 713
 Lin, Zuoquan 618
 Liu, Chunnian 184, 452
 Liu, Hongzhe 626
 Liu, Huan 475
 Liu, Huayong 870
 Liu, Jiming 452
 Liu, Liang Zhang Fangfang 858
 Liu, Lin 913
 Liu, Wei 127, 676
 Liu, Xiaoming 162
 Liu, Xuan 761
 Liu, Yangguang 492
 Liu, Yong 651
 Liu, Zheng 150, 731
 Liu, Zhenyu 245
 Lu, Jianguo 900
 Lu, Jing 172
 Lu, Tijun 761
 Lu, XianLing 919
 Luo, Yi 150
 Lv, Tian-yang 464

 Ma, Lingxiao 882
 Ma, Lisha 404
 McTear, Michael F. 368
 Michener, William K. 610
 Mujeeb-u-Rehman, Maree 538

 Ng, Michael K. 502

 Ou, Weijie 888
 Ou, Zhengyu 566, 725
 Ouyang, Song 804

 Palit, Henry Novianus 556
 Pan, Haiwei 719
 Pan, Heng 670, 919
 Pan, Yunhe 651
 Pan, ZhengYun 919
 Pan, Zhiyong 194
 Pang, Liao-Jun 267
 Park, Byung-Kwon 688
 Park, KwangJin 233

- Parsons, Lance 475
 Pei, Jian 284
 Pennington, Deana D. 610

 Qi, Guoning 840
 Qian, Qian 32, 663
 Qin, Xiangdong 513

 Rasool, Qaisar 416

 Schewe, Klaus-Dieter 737
 Scicchitano, A. 810
 Seo, DongMin 589
 Shao, Zuhua 645
 Shi, Baile 345, 858
 Shi, Pengfei 694
 Shi, Wei 357
 Shi, Yuliang 858
 Shih, N.C. 925
 Shin, Dongil 749
 Shin, Dongkyoo 749
 Son, Minwoo 749
 Song, Hyung Gi 743
 Song, Il-Yeol 688
 Song, MoonBae 233
 Song, SeokIl 589
 Song, Weiwei 767
 Su, Tai-xue 464
 Su, Yila 452
 Sun, Weiwei 345
 Sun, Xiaohua 822
 Sun, Yong 694
 Sun, Yu 731
 Sun, Z.H. 894

 Ta, Na 32, 663
 Tagarelli, A. 810
 Tang, Lv-an 194
 Tang, Shiwei 138, 194, 308, 846
 Thalheim, Bernhard 737
 Tierney, Brendan 906
 Tseng, Lendle 925
 Tu, Kewei 172

 Viglas, Stratis D. 404

 Wang, Chaokun 45
 Wang, Daling 56, 785
 Wang, H.G. 913
 Wang, Hongzhi 68
 Wang, Huayong 81
 Wang, Hui 368
 Wang, Jianjun 846
 Wang, Jilin 638
 Wang, Ju 900
 Wang, Shan 209
 Wang, Shengrui 900
 Wang, Shuxun 767
 Wang, Wei 68, 150, 345
 Wang, Xiaoyuan 345
 Wang, Yi 828
 Wang, Yongheng 706
 Wang, Yuan 102, 578
 Wang, Zheng-xuan 464
 Wei, DaWei 670
 Wen, Quan 767
 Wen, Xuezhi 731
 Wu, Di 601
 Wu, Shanshan 566, 725

 Xia, Hong 297
 Xia, Tian 779
 Xing, Yunpeng 8
 Xiong, Zhongmin 392
 Xu, Congfu 651
 Xu, Dawei 618
 Xu, Hongwei 626
 Xu, Jian 357
 Xu, Jun 502
 Xu, Xiaofei 632
 Xue, Gui-Rong 127, 676
 Xue, Ming 846

 Yang, Donghua 416
 Yang, Dongqing 194, 308, 846
 Yang, Genxing 245
 Yang, Huaizhou 297
 Yang, Shuqiang 706
 Yang, Xiaochun 725
 Yang, Xinhua 566
 Yao, Weipeng 882
 Ye, Jianye 284
 Ye, Wei 245
 Yin, Jianwei 162, 538, 882
 Ying, Ying 194
 Yoo, JaeSoo 589
 Yu, Ge 56, 566, 725, 785
 Yu, Jeffrey Xu 150
 Yu, Jiehua 626
 Yu, Junqing 888

- Yu, Songmei 791
 Yu, Yaxin 566, 725
 Yu, Yijiao 834
 Yu, Yong 127, 172, 676, 864

 Zeng, Hua-jun 127
 Zhang, C.Y. 894
 Zhang, Da-lu 682
 Zhang, Deyun 428
 Zhang, Hui 870
 Zhang, Jianhong 638
 Zhang, Jianting 610
 Zhang, Lei 172
 Zhang, Liang 284
 Zhang, LuWei 913
 Zhang, Meng 785
 Zhang, Ming 308
 Zhang, Qian 731
 Zhang, Qing 345
 Zhang, Quan 816
 Zhang, Shaohua 816
 Zhang, Shutao 257
 Zhang, Wei 719

 Zhang, Xia 731
 Zhang, Yan 513
 Zhang, Zhi 694
 Zhao, Chen 618
 Zhao, Siyang 906
 Zheng, Jun 333
 Zhong, Ning 452
 Zhongdong, Huang 755
 Zhou, Aoying 779
 Zhou, Li-zhu 32, 828
 Zhou, Xiangdong 284
 Zhou, Xiaofang 91, 779
 Zhu, Miaoliang 601
 Zhu, Wang-bin 713
 Zhu, Weibin 864
 Zhu, Xing 864
 Zhu, Yiwen 816
 Zhu, YueFei 670, 919
 Zhuge, Hai 8
 Zong, Yuwei 816
 Zumpano, E. 810
 Zuo, Wan-li 464